



PREPARATION
A LA
CERTIFICATION
LPIC-1
COURS LPI 102

Sommaire

THEME 105 Le noyau

105.1 Gestion dynamique du noyau et des modules (4)

Services et modules noyau

1. Présentation

Le noyau est le cœur du système d'exploitation Linux. Linux en tant que tel est uniquement le nom du noyau développé à l'origine par Linus Torvalds. Le système d'exploitation Linux est composé du noyau et des outils d'exploitation de base.

Le noyau de Linux est libre. Ses sources sont disponibles. Il est donc possible de le recompiler pour l'adapter finement à ses besoins, de le modifier, d'y rajouter des extensions.

Le noyau Linux fait partie de la famille des noyaux monolithiques. C'est-à-dire que toutes ses fonctionnalités et composants sont regroupés dans un programme unique. Cependant depuis la version 2.0 (ou plutôt la version de développement 1.3 pour être plus précis) le noyau est modulaire.

Le noyau est appelé kernel. Il est présent dans /boot et son nom, par convention, commence souvent par vmlinuz-X.Y.Z.p-Vtxt.

On obtient la version du noyau avec la commande uname.

$ uname -r

2.4.9-e.57smp

Les lettres ont une signification particulière.

X : version majeure du noyau. Entre la version 1 et la version 2, le passage au fonctionnement modulaire a été déterminant, ainsi que la réimplémentation de la couche réseau.

Y : une valeur paire représente une branche stable du noyau. Une version impaire représente une branche de développement (attention !). Chaque incrément pair (0,2,4,6) représente une évolution importante du noyau.



La version 2.6 ne dispose pas encore de branche de développement car elle évolue trop vite. Les développeurs ont décidé d'implémenter leurs nouveautés directement dans la version stable.

Z : version mineure du noyau. Quand un lot de modifications par rapport à une version précédente nécessite la diffusion d'un nouveau noyau, alors on incrémente ce chiffre. Par

exemple, un lot regroupant une modification du système son (Alsa qui passe de 1.0.8 à 1.0.9), du système de fichier (ajout de ReiserFS 4), et ainsi de suite...

p : version corrigée ou intermédiaire présente depuis la version 2.6. Quand le noyau nécessite une mise à jour mineure (correction d'un ou deux bugs, etc.) mais pas ou peu d'ajouts de fonctionnalités, on incrémente cette valeur.

V : comme pour les packages, version propre à l'éditeur de la distribution.

txt : on rajoute parfois un texte pour donner des précisions sur le noyau. Par exemple, smp indique un noyau multi-processeur.

2. uname

La commande uname (unix name) permet d'obtenir toutes les informations concernant la version d'Unix (de Linux ici), de manière précise et/ou complète.

Paramètre	Résultat
-m (machine)	Type matériel de la machine.
-n (nodename)	Nom d'hôte de la machine.
-r (release)	Version (numéro) du noyau.
-s (system name)	Nom du système d'exploitation. Par défaut.
-p (processor)	Type de processeur.
-i	Plate-forme matérielle.
-v (version)	Version du système.
-a (all)	Toutes les informations.

$ uname

Linux

$ uname -m

X86_64

$ uname -n

slyserver

$ uname -r

2.6.22.17-0.1-default

$ uname -s

Linux

$ uname -p

$ uname -i

X86_64

$ uname -o

GNU/Linux

$ uname -v

#1 SMP 2008/02/10 20:01:04 UTC

$ uname -a

Linux slyserver 2.6.22.17-0.1-default #1 SMP 2008/02/10 20:01:04 UTC

x86_64 intel x86_64 GNU/Linux

3. Gestion des modules

Les composants de base (scheduler, gestion de la mémoire, des processus, API, etc.) sont toujours présents au sein d'un programme unique. Mais certains pilotes de périphériques, systèmes de fichiers, extensions, protocoles réseaux, etc. peuvent être présents sous forme de modules. Les modules communiquent avec le noyau via une API commune. Ils s'exécutent dans l'espace du noyau. Ils sont paramétrables. Ils peuvent être chargés et déchargés à la demande évitant ainsi un redémarrage de la machine. L'ajout d'un nouveau module (depuis ses sources par exemple) ne nécessite pas de redémarrage.

Les modules sont présents dans /lib/modules/$(uname -r).

cd /lib/modules/$(uname -r)

pwd

/lib/modules/2.6.22.17-0.1-default

Les modules ont un nom qui finit par « .ko » pour kernel object. La terminaison d'origine était « .o » pour les noyaux 2.0 à 2.4. Il s'agit bien de cela : des fichiers objets dynamiquement liés (linked) à leur chargement au noyau, proposant ainsi une API supplémentaire.

cd /lib/modules/$(uname -r)/kernel/fs/vfat

file vfat.ko

vfat.ko: ELF 32-bit LSB relocatable, Intel 80386, version 1 (SYSV),

not stripped

Le mot « relocatable » indique que vous êtes en présence d'un fichier objet.

a. lsmod

La commande lsmod liste les modules actuellement chargés, avec leurs dépendances éventuelles.

```
# lsmod

Module                  Size  Used by

fglrx              1482380  70

nls_iso8859_1          8192  1

nls_cp437              9856  1

vfat                  16128  1

fat                   52636  1 vfat

snd_pcm_oss           50432  0

snd_mixer_oss         20096  1 snd_pcm_oss

snd_seq               54452  0

snd_seq_device        12172  1 snd_seq

iptable_filter         6912  0

ip_tables             16324  1 iptable_filter

ip6_tables            17476  0

x_tables              18308  2 ip_tables,ip6_tables

cpufreq_conservative   11272  0

cpufreq_userspace      8704  0

…
```

La première colonne indique le nom du module chargé. Son nom reflète bien souvent ce à quoi il sert. La seconde colonne donne la taille du module. La troisième colonne fournit un compteur d'utilisation (combien de composants du système accèdent aux modules). La dernière colonne fournit la liste des modules utilisant (donc dépendant) du premier.

Dans l'exemple précédent le module fat est utilisé par le module vfat.

lsmod ne fait que remettre en forme le contenu du fichier virtuel /proc/modules.

cat /proc/modules

fglrx 1482380 60 - Live 0xf9a55000 (P)

nls_iso8859_1 8192 0 - Live 0xf94d0000

nls_cp437 9856 0 - Live 0xf94ed000

snd_pcm_oss 50432 0 - Live 0xf9887000

snd_mixer_oss 20096 1 snd_pcm_oss, Live 0xf94d3000

snd_seq 54452 0 - Live 0xf94de000

snd_seq_device 12172 1 snd_seq, Live 0xf94b9000

…

b. depmod

La commande depmod met à jour l'arbre des dépendances entre les modules en modifiant le fichier modules.dep.

Le fichier /lib/modules/$(uname -r)/modules.dep contient deux colonnes. La première est le chemin du module, la seconde est la liste des dépendances : les modules qui doivent aussi être chargés pour que le premier fonctionne. Voici l'exemple de la ligne correspondant au module vfat :

grep vfat modules.dep

/lib/modules/2.6.22.17-0.1-default/kernel/fs/vfat/vfat.ko:

/lib/modules/2.6.22.17-0.1-default/kernel/fs/fat/fat.ko

Le module vfat.ko dépend du module fat.ko. Il faut donc que le module fat.ko soit chargé en premier pour que vfat.ko fonctionne.

L'usage le plus courant de depmod est avec le paramètre -a qui reconstruit les dépendances de tous les modules correspondant au noyau actuel. Cette action est exécutée à chaque démarrage du système, mais si vous compilez et/ou installez de nouveaux modules, vous devez relancer à la main cette commande pour prendre en compte les nouvelles dépendances.

depmod -a

c. modinfo

La commande modinfo fournit toutes les informations nécessaires sur un module :

le nom du fichier correspondant,

une description du module,

son auteur,

sa licence,

ses dépendances,

ses paramètres,

ses alias matériels.

Les modules ne fournissent pas tous ces informations. Certains modules n’ont pas de paramètres. Dans le premier exemple, le module vfat n’a pas de paramètres, ceux-ci étant mis en place comme options de montage.

modinfo vfat

filename: /lib/modules/2.6.22.17-0.1-default/kernel/fs/fat/fat.ko

license: GPL

srcversion: 886B1B65F96E53415B5811C

depends:

supported: yes

vermagic: 2.6.22.17-0.1-default SMP mod_unload 586

Dans ce deuxième exemple, le module dispose de paramètres. Ce module permet d’utiliser une Webcam contenant une puce ov511. Les paramètres sont volontairement tronqués.

modinfo ov511

filename: /lib/modules/2.6.22.17-0.1-

default/kernel/drivers/media/video/ov511.ko

license: GPL

description: ov511 USB Camera Driver

author: Mark McClelland <mark@alpha.dyndns.org> & Bret Wallach

& Orion Sky Lawlor <olawlor@acm.org> & Kevin Moore & Charl P. Botha

<cpbotha@ieee.org> & Claudio Matsuoka <claudio@conectiva.com>

srcversion: E0DC673BFADEA96F2DED84F

alias: usb:v0813p0002d*dc*dsc*dp*ic*isc*ip*

alias: usb:v05A9pA518d*dc*dsc*dp*ic*isc*ip*

alias: usb:v05A9p0518d*dc*dsc*dp*ic*isc*ip*

alias: usb:v05A9pA511d*dc*dsc*dp*ic*isc*ip*

alias: usb:v05A9p0511d*dc*dsc*dp*ic*isc*ip*

depends: compat_ioctl32,videodev,usbcore

supported: yes

vermagic: 2.6.22.17-0.1-default SMP mod_unload 586

parm: autobright:Sensor automatically changes brightness (int)

parm: autogain:Sensor automatically changes gain (int)

parm: autoexp:Sensor automatically changes exposure (int)

parm: debug:Debug level: 0=none, 1=inits, 2=warning, 3=config,

4=functions, 5=max (int)

parm: snapshot:Enable snapshot mode (int)

parm: cams:Number of simultaneous cameras (int)

parm: compress:Turn on compression (int)

parm: led:LED policy (OV511+ or later). 0=off, 1=on (default),

2=auto (on when open) (int)

Un même module peut gérer plusieurs types de matériel. Il existe plusieurs Webcams disposant d'une puce ov511 ou affiliée pouvant être gérée par le module ov511. Pour que le noyau sache quel module charger lors de la détection de la webcam, ou pour que le module sache quel type de matériel il doit gérer lors de son chargement, il dispose d'alias matériels, usb, pci, scsi, etc., permettant de reconnaître les périphériques qu'il doit gérer.

Les paramètres sont passés au module via les commandes insmod ou modprobe, ou à l'aide du fichier /etc/modprobe.conf.

d. insmod

La commande insmod charge un module donné sans gérer les dépendances. Elle prend en paramètre un nom de module, avec son éventuel chemin. C’est à vous de gérer l’ordre de chargement des modules pour éviter des erreurs liées à des symboles non résolus causées par un problème de dépendance.

Dans l’exemple suivant, les modules fat et vfat n’étant pas chargés, une tentative a lieu pour charger le module vfat.ko seul. Une erreur se produit car ce module dépend de la présence de fat.ko. Le retour de dmesg est éloquent à ce sujet.

lsmod|grep fat

ls

vfat.ko

insmod vfat.ko

insmod: error inserting ’vfat.ko’: -1 Unknown symbol in module

dmesg | tail -20

…

vfat: Unknown symbol fat_dir_empty

vfat: Unknown symbol fat_fs_panic

vfat: Unknown symbol fat_get_dotdot_entry

vfat: Unknown symbol fat_free_clusters

vfat: Unknown symbol fat_scan

vfat: Unknown symbol fat_date_unix2dos

vfat: Unknown symbol fat_search_long

vfat: Unknown symbol fat_getattr

vfat: Unknown symbol fat_attach

vfat: Unknown symbol fat_build_inode

vfat: Unknown symbol fat_fill_super

vfat: Unknown symbol fat_alloc_new_dir

vfat: Unknown symbol fat_notify_change

vfat: Unknown symbol fat_remove_entries

vfat: Unknown symbol fat_add_entries

vfat: Unknown symbol fat_sync_inode

vfat: Unknown symbol fat_detach

Dans ce second exemple, le module fat est tout d'abord chargé, puis le module vfat. Il n'y a pas d'erreur car toutes les dépendances sont présentes.

cd ../fat

ls

fat.ko

insmod fat.ko

cd ../vfat

insmod vfat.ko

lsmod|grep fat

vfat   16128 0

fat   52636 1 vfat

Vous devriez envisager le paramètre -k qui permet l'autoclean des modules. Cette option indique au système de décharger automatiquement un module (compteur à zéro) s'il n'est plus utilisé, permettant de gagner quelques ressources et d'obtenir un système plus propre.

Pour transmettre des paramètres au module, indiquez ceux-ci à la suite de la commande. Par exemple pour le module ov511, vous souhaitez que la led soit active uniquement lorsque la webcam est active et que la compression est activée (pour atteindre les 25 fps annoncés) :

insmod ov511.ko led=2 compress=1

e. rmmod

La commande rmmod décharge le module donné. C'est l'inverse de insmod et, comme cette dernière, rmmod ne gère pas les dépendances :

Il n'est pas possible de décharger un module en cours d'utilisation.

Il n'est pas possible de décharger un module s'il est utilisé par un autre module, même si ce dernier n'est pas utilisé (problème de dépendance).

Dans cet exemple, une clé USB contenant un système de fichiers vfat est branchée et montée. Une première tentative de supprimer vfat échoue.

mount | grep vfat

/dev/sdb1 on /media/disk type vfat …

rmmod vfat

ERROR: Module vfat is in use

Dans ce deuxième exemple, la clé est débranchée. Les modules fat et vfat sont devenus inutiles. On tente de supprimer le module fat. Le système retourne une erreur liée aux dépendances.

rmmod fat

ERROR: Module fat is in use by vfat

Dans ce dernier exemple, le module vfat est déchargé, puis le module fat, dans cet ordre.

rmmod vfat

rmmod fat

f. modprobe

La commande modprobe charge le module donné ainsi que toutes ses dépendances et des paramètres contenus dans /etc/modprobe.conf. Le paramètre -r permet de décharger un modules et ceux qui en dépendent (s'ils ne sont pas utilisés).

Le chargement du module vfat à l'aide de modprobe va automatiquement charger le module fat.

lsmod|grep fat

modprobe vfat

lsmod|grep fat

vfat   16128 0

fat   52636 1 vfat

Maintenant, voyant que seul vfat utilise fat (compteur à 1) mais que rien n'utilise vfat (compteur à zéro), vous pouvez tenter de décharger vfat et les modules dont il dépend s'ils ne sont plus utilisés.

modprobe -r vfat

lsmod|grep fat



Vous pouvez passer des options au module, dans ce cas vous devez vérifier si une ligne « install » correspondant au module existe dans le fichier modprobe.conf et la modifier pour qu'elle accepte les paramètres, ou utiliser une ligne « options ».

g. modprobe.conf

La configuration des modules est placée dans /etc/modprobe.conf. Dans ce fichier vous pouvez définir des alias de modules (très pratique pour les cartes réseau), passer des options aux modules, ajouter des actions au chargement et déchargement d'un module.

Alias et options

Exemple : vous avez deux cartes réseaux. Le pilote de la première carte est contenu dans le module e1000. Vous voulez qu'il soit chargé en utilisant le nom eth0. Le pilote de la seconde carte est dans le module airo et vous souhaitez le charger en utilisant le nom eth1. Vous allez aussi spécifier des paramètres au module ov511 dont vous avez récupéré les informations avec modinfo.

cat /etc/modprobe.conf

...

alias eth0 e1000

alias eth1 airo

options ov511 led=2 compress=1

...

Ensuite vous chargez les modules soit grâce à leur nom, soit grâce à leur alias. Dans tous les cas, les paramètres sont automatiquement pris en compte.

modprobe eth0

modprobe eth1

modprobe ov511

Les options des modules peuvent aussi être utilisés avec les alias.

alias webcam ov511

options webcam led=2 compress=1

...

install et remove

Les commandes install et remove du fichier modprobe.conf sont les plus puissantes. Lors du chargement d'un module, si modprobe trouve une commande install associée, il ne charge pas le module mais exécute les commandes indiquées. La commande peut être ce que vous voulez, comme par exemple le chargement d'un module plus adapté, la mise en place d'une configuration spécifique, l'exécution d'un script, etc.

L'exemple suivant tente de charger le module ov511_new s'il existe, sinon il bascule sur ov511, lorsque vous invoquez modprobe webcam.

install webcam /sbin/modprobe ov511_new || /sbin/modprobe ov511

modprobe peut prendre un paramètre pratique lui permettant d'ignorer les lignes install de modprobe.conf : --ignore-install. L'exemple suivant charge le module ahci puis le module ata_piix lorsque vous tentez de charger ce dernier. Sans le paramètre, modprobe tournerait en boucle, relisant et interprétant la ligne install à chaque tentative de chargement de ata_piix.

install ata_piix /sbin/modprobe ahci 2>&1 |:; /sbin/modprobe --

ignore-install ata_piix

La commande remove fait la même chose, mais au déchargement du module avec modprobe -r.

CMDLINE_OPTS

Si une ligne install est présente dans modprobe.conf et que vous tentez de charger le module correspondant avec modprobe et des paramètres, ces derniers ne seront pas pris en compte sauf si vous avez rajouté derrière le module la chaîne $CMDLINE_OPTS.

install webcam /sbin/modprobe ov511_new $CMDLINE_OPTS ||

/sbin/modprobe ov511 $CMDLINE_OPTS

4. Chargement des modules au boot

a. initrd

Certains modules peuvent être nécessaires au démarrage de la machine, notamment pour monter un système de fichiers. Comment monter la partition racine en ext3 alors que le module gérant ce type de système de fichiers est sur cette partition (et pas en dur dans le noyau) ? Ces modules sont placés dans une image de disque mémoire initiale ou initrd (initial ramdisk). Ces fichiers compressés sont chargés au démarrage en mémoire et sont des ramdisks. Ils contiennent des instructions et modules qui sont chargés au démarrage.

$ ls -l /boot/initrd*

-rw-r--r-- 1 root root 4151529 fév 13 10:48 /boot/initrd-2.6.22.17-0.1-

default

file /boot/initrd-2.6.22.17-0.1-default

/boot/initrd-2.6.22.17-0.1-default: gzip compressed data, from Unix,

last modified: Wed Feb 13 10:48:34 2008, max compression

Il est possible d’extraire le contenu de initrd et même de le modifier et le reconstruire pour l’adapter. Ce fichier est une archive cpio compressée avec gzip.

zcat /boot/initrd-2.6.22.17-0.1-default | cpio -id --no-absolute-

filenames

17588 blocks

p64p17bicb3:/home/seb/initrd # ls -l

total 196

drwxr-xr-x 2 root root   4096 avr 2 12:22 bin

drwxr-xr-x 2 root root   4096 avr 2 12:22 boot

-rw-r--r-- 1 root root 133490 avr 2 12:22 bootsplash

drwxr-xr-x 2 root root   4096 avr 2 12:22 config

drwxr-xr-x 2 root root   4096 avr 2 12:22 dev

drwxr-xr-x 6 root root   4096 avr 2 12:22 etc

-rwxr-xr-x 1 root root   2067 avr 2 12:22 init

drwxr-xr-x 5 root root   4096 avr 2 12:22 lib

drwxr-xr-x 2 root root   4096 avr 2 12:22 proc

drwxr-xr-x 2 root root   4096 avr 2 12:22 root

-rw-r--r-- 1 root root   3641 avr 2 12:22 run_all.sh

drwxr-xr-x 2 root root   4096 avr 2 12:22 sbin

drwxr-xr-x 2 root root   4096 avr 2 12:22 sys

drwsrwxrwx 2 root root   4096 avr 2 12:22 tmp

drwxr-xr-x 3 root root   4096 avr 2 12:22 usr

drwxr-xr-x 5 root root   4096 avr 2 12:22 var

Quand initrd est chargé et monté, le noyau tente d’exécuter le script init présent à la racine du pseudo système de fichiers. C’est lui qui doit charger les modules nécessaires et établir la toute première configuration de base, avant que le noyau ne monte le système de fichier racine et exécute /sbin/init. Dans la pratique, c’est ce fichier (ou l’un de ceux qu’il appelle) qui monte le système de fichiers racine et qui passe la main à /sbin/init.



Sur certaines distributions, notamment anciennes, c’est un script appelé /sbin/init qui était appelé. Ceci a été supprimé pour éviter de confondre le script /sbin/init de initrd avec le « vrai » init du système. Si un fichier linuxrc est présent à la racine de l’initrd, il est exécuté. En cas de doute sur le nom du script lancé, celui-ci est dans les sources du noyau, init/main.c, variable ram_execute_command.

$ cd /usr/src/linux

$ grep -n ram_execute_commande init/main.c

...

865:           ramdisk_execute_command = "/init";

...

La commande mkinitrd permet de reconstruire un fichier initrd standard, mais il est bien souvent possible, selon la distribution, de lui fournir une liste de modules à y placer et donc à charger :

Sous Red Hat, c’est l’option --preload=module, qui peut être utilisée plusieurs fois, qui précise quels sont les modules à charger.

Sous openSUSE, c’est l’option -m "module1 module2 etc." qui précise cette liste. Voyez cependant la méthode utilisant les variables de sysconfig.

Sous Debian, il faut placer les noms des modules dans le fichier /etc/mkinitrd/modules.

Voici le résultat d’une commande mkinitrd sous openSUSE. La ligne « Kernel Modules » précise les modules chargés par initrd.

mkinitrd

Kernel image: /boot/vmlinuz-2.6.22.17-0.1-default

Initrd image: /boot/initrd-2.6.22.17-0.1-default

Root device: /dev/disk/by-id/scsi-SATA_ST380011A_5JVTH798-part6

(/dev/sda6) (mounted on / as ext3)

Resume device: /dev/sda5

Kernel Modules: processor thermal scsi_mod libata sata_sil

pata_atiixp fan jbd mbcache ext3 edd sd_mod usbcore ohci-hcd uhci-hcd

ehci-hcd ff-memless hid usbhid

Features: block usb resume.userspace resume.kernel

Bootsplash: SuSE (1280x1024)

17588 blocks

Un initrd bien construit (ou plutôt une commande mkinitrd correcte) recopie aussi la commande modprobe et le fichier de configuration modprobe.conf qui peut (et doit) contenir les paramètres des modules si besoin.

b. Red Hat /etc/rc.modules

La méthode préférée sous Red Hat est de charger les modules depuis l'initrd. Cependant vous avez la possibilité de créer un script /etc/rc.modules qui contiendra les commandes nécessaires au chargement des modules, principalement avec modprobe comme vu précédemment. Ce fichier est exécuté par rc.sysinit au démarrage du système, via l'inittab. Il doit être rendu exécutable.

c. openSUSE : /etc/sysconfig/kernel

La distribution openSUSE place sa configuration dans une arborescence /etc/sysconfig. Le fichier /etc/sysconfig/kernel contient quelques éléments de configuration du noyau mais surtout deux variables chargées de spécifier les modules à charger :

INITRD_MODULES est la liste des modules chargés par l'initrd.

MODULES_LOADED_ON_BOOT est la liste des modules chargé par init (donc après initrd).

Vous verrez une différence entre la liste des modules indiquée pour l'initrd et la liste réelle chargée lors de la création de celui-ci. La commande mkinitrd de openSUSE permet de spécifier des fonctionnalités préétablies lors de la création de l'initrd (option -f "feature1 feature2 etc"). La fonctionnalité « usb » va par exemple charger tous les modules liés au

support USB, ce qui est vital si vous disposez d'un clavier USB ou si vous bootez depuis une clé ou un disque externe.

d. Debian : /etc/modules

Sous Debian il suffit de rajouter les noms des modules à charger dans le fichier /etc/modules. Après le nom des modules vous pouvez indiquer des paramètres. Pour chaque ligne un modprobe est exécuté. Vous pouvez rajouter un commentaire après un #. C'est le fichier /etc/init.d/modutils qui charge les modules au démarrage via init.

```
# /etc/modules: kernel modules to load at boot time.

#

# This file should contain the names of kernel modules that are

# to be loaded at boot time, one per line.  Comments begin with

# a "#", and everything on the line after them are ignored.



ide-cd

ide-detect

ov511
```

5. Paramètres dynamiques

a. /proc et /sys

/proc et /sys sont des systèmes de fichiers virtuels contenant des informations sur le noyau en cours d'exécution. La version 2.4 du noyau ne connaît que /proc où toutes les informations sont regroupées. La version 2.6 du noyau a modifié la fonction de /proc pour déléguer une partie de ses tâches à /sys.

S'agissant de systèmes de fichiers virtuels, ils ne prennent aucune place ni en mémoire, ni sur un disque quelconque. Il ne faut pas se laisser berner par la taille des pseudo-fichiers contenus dedans. Ne tentez pas de supprimer /proc/kcore pour gagner de la place ! Tous ces fichiers (ou presque) peuvent être lus et affichés directement.

Les fichiers de /proc vous donneront énormément d'informations sur le système :

interrupts : les paramètres IRQ.

cpuinfo : détails sur vos processeurs.

dma : les paramètres DMA.

ioports : les ports mémoires E/S.

devices : les périphériques présents.

meminfo : l'état global de la mémoire.

loadavg : la charge du système.

uptime : uptime du système, attente.

version : détails de la version de Linux.

modules : identique au résultat de lsmod.

swaps : liste et état des partitions d'échange.

partitions : liste et état des partitions connues du système.

mounts : montages des systèmes de fichiers.

pci : détails du bus PCI.

$ cat /proc/cpuinfo

processor : 0

vendor_id : GenuineIntel

cpu family : 6

model : 8

model name : Pentium III (Coppermine)

stepping : 6

cpu MHz : 996.895

cache size : 256 KB

physical id : 1360587528

siblings : 1

fdiv_bug : no

hlt_bug : no

f00f_bug : no

coma_bug : no

fpu : yes

fpu_exception : yes

cpuid level : 2

wp : yes

flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr

pge mca cmov pat pse36 mmx fxsr sse

bogomips : 1985.74

$ cat /proc/version

Linux version 2.4.9-e.57smp (bhcompile@tweety.build.Red Hat.com) (gcc

version 2.96 20000731 (Red Hat Linux 7.2 2.96-129.7.2)) #1 SMP Thu

Dec 2 20:51:12 EST 2004

/proc contient des sous-répertoires qui regroupent des informations par thème.

/proc/scsi : informations sur le bus SCSI.

/proc/ide : informations sur le bus IDE.

/proc/net : informations sur le réseau.

/proc/sys : paramètres et configuration dynamique du noyau.

/proc/<PID> : informations sur le processus PID.

Certaines entrées des systèmes de fichiers /proc/sys et /sys sont différentes des autres car leur contenu peut être modifié et les modifications sont prises en compte directement par le noyau sans avoir à redémarrer la machine.

Par exemple voici comment activer le forwarding IP et passer le nombre de handles de fichiers de 8192 à 16384.

echo "1" > /proc/sys/net/ipv4/ip_forward

echo "16384" > /proc/sys/fs/file-max

b. sysctl

Les valeurs modifiées ne sont pas enregistrées. En cas de redémarrage il faut recommencer. Le fichier rc.sysinit appelle la commande sysctl qui agit sur ces paramètres. Pour que les valeurs restent permanentes (remises en place à chaque démarrage) il faut modifier le fichier /etc/sysctl.conf. Vous pouvez rechercher les modifications manuellement.

sysctl -e -p /etc/sysctl.conf

sysctl -a

...

dev.raid.speed_limit_max = 100000

dev.raid.speed_limit_min = 100

net.token-ring.rif_timeout = 60000

net.ipv4.conf.eth1.arp_filter = 0

net.ipv4.conf.eth1.tag = 0

net.ipv4.conf.eth1.log_martians = 0

net.ipv4.conf.eth1.bootp_relay = 0

net.ipv4.conf.eth1.proxy_arp = 0

net.ipv4.conf.eth1.accept_source_route = 1

net.ipv4.conf.eth1.send_redirects = 1

net.ipv4.conf.eth1.rp_filter = 1

net.ipv4.conf.eth1.shared_media = 1

net.ipv4.conf.eth1.secure_redirects = 1

...

Consulter les traces du système

1. dmesg

La commande **dmesg** permet de récupérer les messages du noyau émis au démarrage de la machine, puis les messages émis par la suite. Le tampon de dmesg est circulaire. Au bout d'un certain nombre de messages, les premiers disparaissent. Cependant ces traces ne sont pas perdues car le service syslogd (chapitre Le réseau) écrit ces éléments dans des fichiers.

Cette commande est généralement la première lancée par un administrateur, ingénieur ou exploitant du système Linux pour vérifier la présence d'éventuelles erreurs. En effet, après le boot les messages continuent d'arriver, notamment lors de la connexion à chaud de périphériques, au chargement de certains modules, lorsque des crashs se produisent, lors d'une corruption du système de fichiers, etc.

L'exemple suivant est volontairement tronqué à une cinquantaine de lignes, la sortie originale en contenant plus de 500. Le début représente le tout début de l'exécution du noyau (informations fournies par le BIOS). Le milieu montre la détection du premier disque dur et de ses partitions. La fin montre ce qu'il se passe à l'insertion d'une clé USB, après le boot au cours d'une utilisation normale, et à sa déconnexion.

```
# dmesg

Linux version 2.6.22.17-0.1-default (geeko@buildhost) (gcc version 4.2.1

(SUSE Linux)) #1 SMP 2008/02/10 20:01:04 UTC

BIOS-provided physical RAM map:

 BIOS-e820: 0000000000000000 - 000000000009fc00 (usable)

 BIOS-e820: 000000000009fc00 - 00000000000a0000 (reserved)

 BIOS-e820: 00000000000e4000 - 0000000000100000 (reserved)

 BIOS-e820: 0000000000100000 - 000000003bfd0000 (usable)

 BIOS-e820: 000000003bfd0000 - 000000003bfde000 (ACPI data)

 BIOS-e820: 000000003bfde000 - 000000003c000000 (ACPI NVS)

 BIOS-e820: 00000000ff780000 - 0000000100000000 (reserved)

63MB HIGHMEM available.

896MB LOWMEM available.

found SMP MP-table at 000ff780

…

(quelques dizaines de lignes)
```

…

sd 4:0:0:0: [sda] 156301488 512-byte hardware sectors (80026 MB)

sd 4:0:0:0: [sda] Write Protect is off

sd 4:0:0:0: [sda] Mode Sense: 00 3a 00 00

sd 4:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA

sda:<6>ehci_hcd 0000:00:13.2: new USB bus registered, assigned bus number 2

ehci_hcd 0000:00:13.2: irq 19, io mem 0xff6fc000

sda1 sda2 < sda5 sda6 sda7 >

sd 4:0:0:0: [sda] Attached SCSI disk

…

(quelques dizaines de lignes)

…

usb-storage: device scan complete

usb 2-6: new high speed USB device using ehci_hcd and address 8

usb 2-6: new device found, idVendor=126f, idProduct=0161

usb 2-6: new device strings: Mfr=0, Product=2, SerialNumber=3

usb 2-6: Product: USB Mass Storage Device

usb 2-6: SerialNumber: 0c1738a65e944d

usb 2-6: configuration #1 chosen from 1 choice

scsi10 : SCSI emulation for USB Mass Storage devices

usb-storage: device found at 8

usb-storage: waiting for device to settle before scanning

scsi 10:0:0:0: Direct-Access   USB2.0   Mobile Disk   1.00 PQ:

0 ANSI: 2

sd 10:0:0:0: [sdc] 1007616 512-byte hardware sectors (516 MB)

sd 10:0:0:0: [sdc] Write Protect is off

sd 10:0:0:0: [sdc] Mode Sense: 00 00 00 00

sd 10:0:0:0: [sdc] Assuming drive cache: write through

sd 10:0:0:0: [sdc] 1007616 512-byte hardware sectors (516 MB)

sd 10:0:0:0: [sdc] Write Protect is off

sd 10:0:0:0: [sdc] Mode Sense: 00 00 00 00

sd 10:0:0:0: [sdc] Assuming drive cache: write through

sdc: sdc1

sd 10:0:0:0: [sdc] Attached SCSI removable disk

sd 10:0:0:0: Attached scsi generic sg3 type 0

usb-storage: device scan complete

usb 2-5: USB disconnect, address 7

Pour exploiter le résultat, l'idéal est soit de rediriger celui-ci dans un fichier pour une analyse à froid, soit d'utiliser la commande **grep** à bon escient, si vous savez ce que vous cherchez.

dmesg|grep CPU

Initializing CPU#0

CPU: After generic identify, caps: 078bfbff e3d3fbff 00000000 00000000

00000001 00000000 00000001

CPU: L1 I Cache: 64K (64 bytes/line), D cache 64K (64 bytes/line)

CPU: L2 Cache: 256K (64 bytes/line)

CPU: After all inits, caps: 078bfbff e3d3fbff 00000000 00000410 00000001

00000000 00000001

Intel machine check reporting enabled on CPU#0.

CPU0: AMD Sempron(tm) Processor 3200+ stepping 02

Brought up 1 CPUs

Switched to NOHz mode on CPU #0

2. /var/log/messages

Quelle que soit la distribution employée, **/var/log/messages** est le fichier central des messages du système, qu'ils proviennent du noyau ou des services. Le contenu de ce fichier, géré par syslog, reflète l'état global du système (et pas uniquement du noyau) au cours de son utilisation. Sur un système classique, son contenu reprend celui issu de la commande **dmesg** et celui de divers services.

Les lignes sont horodatées. En effet sans action spéciale (voir logrotate au chapitre Le réseau) le fichier grossit dans le temps et n'est pas purgé. Un fichier messages peut contenir plusieurs milliers de lignes, surtout si un problème survient !

wc -l < messages

8453

Aussi tout comme avec la commande **dmesg**, prenez soin d'effectuer un grep pour sélectionner vos lignes (ou tail, head, etc.).

tail -100 /var/log/messages | grep fglrx

Feb 5 09:12:22 p64p17bicb3 kernel: [fglrx] interrupt source 20008000

successfully disabled!

Feb 5 09:12:22 p64p17bicb3 kernel: [fglrx] enable ID = 0x00000000

Feb 5 09:12:22 p64p17bicb3 kernel: [fglrx] Receive disable interrupt

message with irqEnableMask: 20008000; dwIRQEnableId: 00000004

Feb 5 09:13:32 p64p17bicb3 kernel: [fglrx] Maximum main memory to use

for locked dma buffers: 867 MBytes.

Feb 5 09:13:32 p64p17bicb3 kernel: [fglrx] GART Table is not in FRAME_

BUFFER range

Feb 5 09:13:32 p64p17bicb3 kernel: [fglrx] Reserve Block - 0 offset =

0X0 length = 0X40000

Feb 5 09:13:32 p64p17bicb3 kernel: [fglrx] Reserve Block - 1 offset =

0X3ff5000 length = 0Xb000

Feb 5 09:13:32 p64p17bicb3 kernel: [fglrx] interrupt source 20008000

successfully enabled

Feb 5 09:13:32 p64p17bicb3 kernel: [fglrx] enable ID = 0x00000004

Feb 5 09:13:32 p64p17bicb3 kernel: [fglrx] Receive enable interrupt

message with irqEnableMask: 20008000

TP : Démarrage de Linux, services, noyau et périphériques - Noyau et modules

105.2 Recompiler un noyau personnalisé (3)

Compiler un noyau

1. Obtenir les sources

a. Sources officielles

Les sources officielles du noyau sont disponibles depuis le site kernel.org. Elles portent le nom de vanilla. Un noyau (ou kernel) vanilla est un noyau brut, sans ajout de patches, issu directement des développements des contributeurs du noyau, et n'a pas été adapté à une quelconque distribution.

Ce sont les sources du noyau qui sont fournies. Vous devez configurer, compiler et installer un noyau, et éventuellement créer un initrd avant de pouvoir l'utiliser.

Le noyau est fourni sous forme d'archive compressée que vous devez ouvrir avec les outils adaptés.

$ ls

linux-2.6.24.4.tar.bz2

$ tar xvjf linux-2.6.24.4.tar.bz2

linux-2.6.24.4/

linux-2.6.24.4/.gitignore

linux-2.6.24.4/.mailmap

linux-2.6.24.4/COPYING

linux-2.6.24.4/CREDITS

linux-2.6.24.4/Documentation/

linux-2.6.24.4/Documentation/00-INDEX

linux-2.6.24.4/Documentation/ABI/

linux-2.6.24.4/Documentation/ABI/README

linux-2.6.24.4/Documentation/ABI/obsolete/

linux-2.6.24.4/Documentation/ABI/obsolete/dv1394

linux-2.6.24.4/Documentation/ABI/removed/

linux-2.6.24.4/Documentation/ABI/removed/devfs

linux-2.6.24.4/Documentation/ABI/removed/raw1394_legacy_isochronous

linux-2.6.24.4/Documentation/ABI/stable/

linux-2.6.24.4/Documentation/ABI/stable/syscalls

linux-2.6.24.4/Documentation/ABI/stable/sysfs-module

linux-2.6.24.4/Documentation/ABI/testing/

linux-2.6.24.4/Documentation/ABI/testing/debugfs-pktcdvd

linux-2.6.24.4/Documentation/ABI/testing/sysfs-bus-usb

linux-2.6.24.4/Documentation/ABI/testing/sysfs-class

linux-2.6.24.4/Documentation/ABI/testing/sysfs-class-pktcdvd

linux-2.6.24.4/Documentation/ABI/testing/sysfs-devices

linux-2.6.24.4/Documentation/ABI/testing/sysfs-kernel-uids

linux-2.6.24.4/Documentation/ABI/testing/sysfs-power

...

Les sources du noyau sont placées dans **/usr/src/linux**. Si plusieurs versions des sources de noyaux sont présentes, Linux est un lien symbolique vers la source du noyau courant.

ls -l

lrwxrwxrwx 1 root root   19 fév 13 10:54 linux -> linux-2.6.22.17-0.1

drwxr-xr-x 4 root root   4096 fév 13 10:54 linux-2.6.22.16-0.2

drwxr-xr-x 20 root root   4096 avr 7 10:57 linux-2.6.22.17-0.1

drwxr-xr-x 3 root root   4096 fév 13 10:52 linux-2.6.22.17-0.1-obj

drwxrwxr-x 20 root root   4096 mar 24 19:49 linux-2.6.24.4

drwxr-xr-x 6 root root   4096 fév 1 14:37 linux-2.6.24-g13f09b95-15

lrwxrwxrwx 1 root root   23 fév 13 10:54 linux-obj -> linux-2.6.22

.17-0.1-obj

b. Sources de la distribution

Chaque distribution est fournie avec un noyau bien souvent patché. Ces modifications peuvent revêtir plusieurs aspects : ajout de pilotes, backports (mises à jour rétroactives, rajout de fonctionnalités issues d'une version ultérieure) de noyaux plus récents, correctifs de sécurité, rajout de fonctionnalités, etc. Elles sont appliquées sur un noyau vanilla, et souvent dans un ordre précis.

Compiler un noyau n'est pas anodin. Si vous disposez d'un contrat de support avec l'éditeur de la distribution (par exemple pour une version Serveur de Red Hat), et si vous rencontrez des problèmes avec le noyau par défaut, vous devriez plutôt envisager de :

vérifier si une mise à jour officielle corrige votre problème ;

remonter votre problème à la hotline de l'éditeur.

Avant de penser à installer un noyau recompilé par vos soins.

Si vous souhaitez tout de même recompiler le noyau, la démarche est la même que pour les sources officielles. Seulement vous devez installer le package des sources, ce qui devrait avoir pour effet d'installer en même temps tous les outils nécessaires à la compilation.

Le package s'appelle kernel-sources sur openSUSE et Red Hat. Pour ce dernier, rendez-vous sur le site de Red Hat car le package kernel-sources n'est pas nécessaire pour compiler des nouveaux modules (il faut le package kernel-devel) et il n'est pas très simple de les obtenir. Sous Debian le package se nomme linux-source-2.x.y (x et y représentant la version du noyau). Vous devez installer ces packages avec les outils adaptés à votre distribution.

2. Les outils nécessaires

Pour compiler le noyau Linux, il est nécessaire de disposer de certains outils :

le compilateur C ;

les bibliothèques de développement C standard ;

la bibliothèque ncurses (pour menuconfig) ;

la bibliothèque qt3 (pour xconfig) ;

les outils Make ;

les modutils ;

mkinitrd ;

de la concentration ;

des nerfs solides ;

de la patience ;

Ces outils sont fournis en standard avec la distribution, sauf les trois derniers…

3. Configuration

a. Le .config

Avant de lancer la compilation du noyau, vous devez sélectionner les options, fonctionnalités et pilotes à conserver ou non. Ceci se fait par divers moyens exposés par la suite. Cette configuration est ensuite sauvée dans un fichier **/usr/src/linux/.config**. Ce fichier contient un grand nombre de variables, chacune d'entre elles correspondant à une option du noyau, et pouvant prendre trois valeurs :

y : la fonctionnalité est présente et intégrée au sein du noyau monolithique, ou, si elle dépend d'un module, intégrée au sein de ce module ;

m : la fonctionnalité sera compilée sous forme de module ;

n : la fonctionnalité est absente.

Dans le dernier cas, cette valeur est rarement présente. Il suffit que l'option soit absente du fichier pour qu'elle ne soit pas activée lors de la compilation. Dans ce cas la ligne correspondante est simplement commentée avec un # devant.

L'exemple suivant montre que l'option MCORE2 (optimisation pour les Intel Core2) n'est pas active.

$ grep "^#" .config | grep -i core2

CONFIG_MCORE2 is not set

L'exemple suivant montre les lignes de configuration associées aux fonctionnalités du système de fichiers ext3. Le support du système de fichier ext3 est compilé sous forme de module (première ligne). Les fonctionnalités supplémentaires de ext3 (attributs étendus, ACL, sécurité, attributs nfs4) dépendent de la première ligne, elles seront donc présentes au sein du module et non dans le noyau, ou comme module à part.

grep -i ext3 .config

CONFIG_EXT3_FS=m

CONFIG_EXT3_FS_XATTR=y

CONFIG_EXT3_FS_POSIX_ACL=y

CONFIG_EXT3_FS_NFS4ACL=y

CONFIG_EXT3_FS_SECURITY=y

Certaines options dépendent d'autres. Sauf si vous savez exactement ce que vous faites, vous ne devriez pas éditer le fichier **.config** à la main pour y effectuer des modifications, au risque de casser des dépendances lors de la compilation. Le plus simple reste de passer par les étapes de configuration exposées par la suite.

Évitez de modifier **.config** à la main, passez plutôt par les outils fournis depuis les sources ou par votre distribution.

b. Récupérer la configuration du noyau

La configuration actuelle du noyau peut être accessible depuis plusieurs endroits. Si un noyau (ou ses sources) provient d'un package de la distribution, il est probable que le fichier .config soit déjà présent au sein de **/usr/src/linux**, ou ailleurs, auquel cas vous vous rapporterez à la documentation officielle.

Sous openSUSE par exemple le répertoire **/boot** contient une copie du .config ayant servi à la compilation du noyau.

ls /boot/config*

/boot/config-2.6.22.17-0.1-default

Dans ce cas vous pouvez réutiliser cette configuration à la main :

cp /boot/config-2.6.22.17-0.1-default /usr/src/linux/.config

Les noyaux sont souvent configurés avec deux options intéressantes.

grep -i KCONF .config

CONFIG_IKCONFIG=y

CONFIG_IKCONFIG_PROC=y

La première permet de placer le contenu du .config dans le noyau lui-même lors de la compilation. La seconde permet d'accéder à cette configuration depuis le système de fichiers virtuel **/proc** via le fichier **/proc/config.gz**. Ce pseudo-fichier est compressé au format gzip. Pour le lire, utilisez la commande **zcat**.

zcat /proc/config.gz > /usr/src/linux/.config

c. make oldconfig

La méthode précédente présente quelques inconvénients notamment lorsqu'il s'agit de récupérer la configuration d'un noyau plus récent ou plus ancien :

des fonctionnalités peuvent avoir disparu ;

d'autres ont pu être ajoutées.

Dans ce cas, partir d'un fichier de configuration inadapté peut avoir des conséquences néfastes. Pour éviter tout problème, le mieux est d'utiliser une possibilité offerte par les sources du noyau (ou plutôt du Makefile) : récupérer l'ancienne configuration, l'analyser, indiquer les changements survenus et demander quoi faire. Ceci se fait via la commande **make oldconfig**.

Dans l'exemple suivant, la configuration d'un noyau 2.6.22 d'origine openSUSE est reprise pour un noyau vanilla 2.6.24. De nombreux avertissements apparaissent (volontairement tronqués) car le noyau vanilla ne contient pas certaines options issues de patches spécifiques au noyau de la distribution. Ensuite (lignes en gras), le nouveau noyau dispose de nouvelles possibilités absentes à l'origine. Vous devez alors répondre à chacune des questions ce qui est assez fastidieux.

make oldconfig

  HOSTCC scripts/basic/fixdep

  HOSTCC scripts/basic/docproc

  HOSTCC scripts/kconfig/conf.o

  HOSTCC scripts/kconfig/kxgettext.o

  SHIPPED scripts/kconfig/zconf.tab.c

SHIPPED scripts/kconfig/lex.zconf.c

SHIPPED scripts/kconfig/zconf.hash.c

HOSTCC scripts/kconfig/zconf.tab.o

HOSTLD scripts/kconfig/conf

scripts/kconfig/conf -o arch/x86/Kconfig

#

using defaults found in /boot/config-2.6.22.17-0.1-default

#

/boot/config-2.6.22.17-0.1-default:36:warning: trying to assign

nonexistent symbol SUSE_KERNEL

/boot/config-2.6.22.17-0.1-default:39:warning: trying to assign

nonexistent symbol IPC_NS

/boot/config-2.6.22.17-0.1-default:47:warning: trying to assign

nonexistent symbol UTS_NS

/boot/config-2.6.22.17-0.1-default:125:warning: trying to assign

nonexistent symbol X86_XEN

/boot/config-2.6.22.17-0.1-default:177:warning: trying to assign

nonexistent symbol X86_MINIMUM_CPU_MODEL

/boot/config-2.6.22.17-0.1-default:252:warning: trying to assign

nonexistent symbol PM_SYSFS_DEPRECATED

/boot/config-2.6.22.17-0.1-default:253:warning: trying to assign

nonexistent symbol SOFTWARE_SUSPEND

…

/boot/config-2.6.22.17-0.1-default:3814:warning: trying to assign

nonexistent symbol KDB

/boot/config-2.6.22.17-0.1-default:3824:warning: symbol value ’m’

invalid for SECURITY_CAPABILITIES

/boot/config-2.6.22.17-0.1-default:3825:warning: symbol value ’m’

invalid for SECURITY_ROOTPLUG

/boot/config-2.6.22.17-0.1-default:3827:warning: trying to assign

nonexistent symbol SECURITY_APPARMOR

*

* Linux Kernel Configuration

*

*

* General setup

*

Prompt for development and/or incomplete code/drivers (EXPERIMENTAL)

[Y/n/?] y

Local version - append to kernel release (LOCALVERSION) [-default]

-default

Automatically append version information to the version string

(LOCALVERSION_AUTO) [N/y/?] n

Support for paging of anonymous memory (swap) (SWAP) [Y/n/?] y

System V IPC (SYSVIPC) [Y/n/?] y

POSIX Message Queues (POSIX_MQUEUE) [Y/n/?] y

BSD Process Accounting (BSD_PROCESS_ACCT) [Y/n/?] y

BSD Process Accounting version 3 file format (BSD_PROCESS_ACCT_V3)

[Y/n/?] y

Export task/process statistics through netlink (EXPERIMENTAL) (TASKSTATS)

[Y/n/?] y

  Enable per-task delay accounting (EXPERIMENTAL) (TASK_DELAY_ACCT)

[Y/n/?] y

  Enable extended accounting over taskstats (EXPERIMENTAL) (TASK_XACCT)

[N/y/?] n

User Namespaces (EXPERIMENTAL) (USER_NS) [N/y/?] (NEW) n

PID Namespaces (EXPERIMENTAL) (PID_NS) [N/y/?] (NEW) n

Auditing support (AUDIT) [Y/n/?] y

  Enable system-call auditing support (AUDITSYSCALL) [Y/n/?] y

Kernel .config support (IKCONFIG) [Y/n/m/?] y

  Enable access to .config through /proc/config.gz (IKCONFIG_PROC)

[Y/n/?] y

Kernel log buffer size (16 => 64KB, 17 => 128KB) (LOG_BUF_SHIFT) [17] 17

Control Group support (CGROUPS) [N/y/?] (NEW) y

  Example debug cgroup subsystem (CGROUP_DEBUG) [N/y/?] (NEW) n

  Namespace cgroup subsystem (CGROUP_NS) [N/y/?] (NEW) n

  Cpuset support (CPUSETS) [Y/n/?] y

Fair group CPU scheduler (FAIR_GROUP_SCHED) [Y/n/?] (NEW) y

…



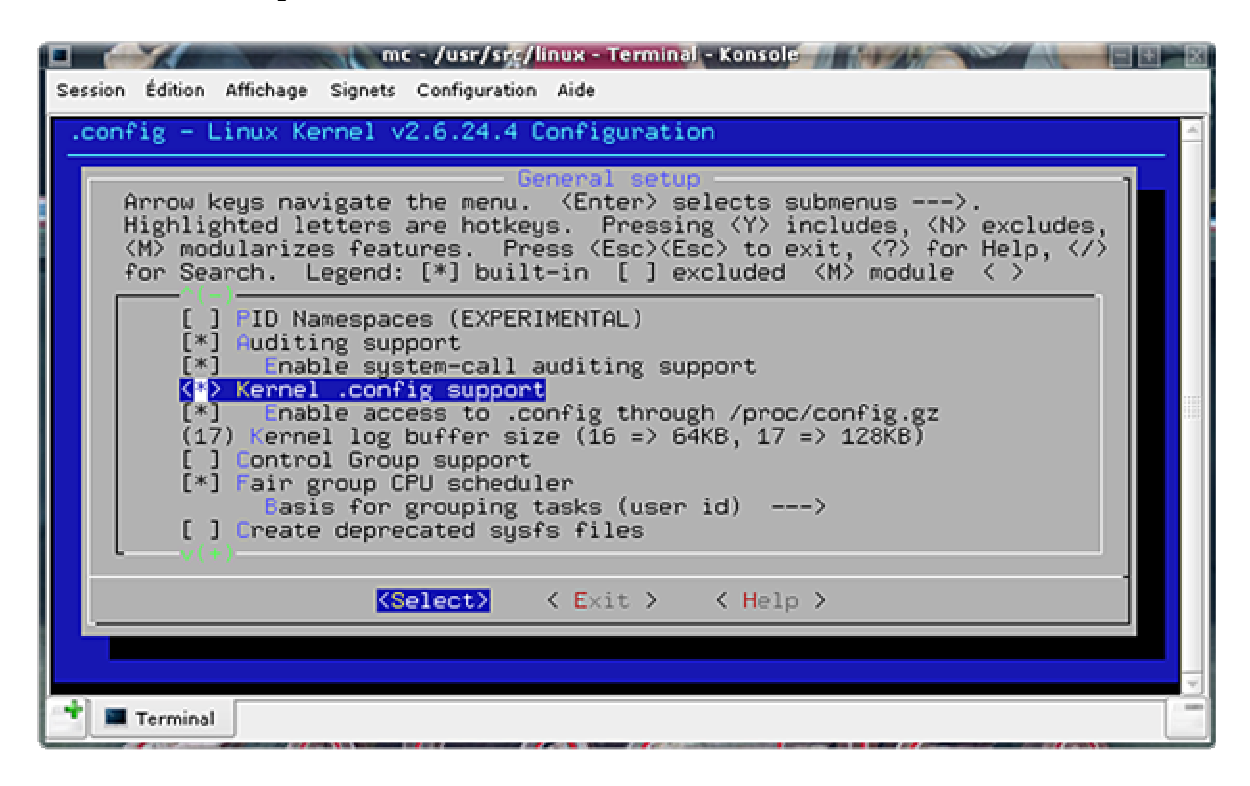
Si vous appuyez sur la touche [Entrée], c'est l'option par défaut qui est activée. Vous pouvez laisser le doigt appuyé sur la touche [Entrée] jusqu'à la fin, puis ensuite modifier la configuration avec l'interface textuelle ou graphique.

d. make menuconfig

La méthode précédente est pratique dans le cas d'une migration. Cependant vous voudrez peut-être utiliser quelque chose de plus convivial. Si vous ne disposez pas d'interface

graphique (elle est souvent absente sur les serveurs car inutile) vous pouvez configurer votre noyau en passant par une interface en mode console. Voilà qui devrait rappeler MS-DOS à certains.

make menuconfig



Les options de compilation du noyau via menuconfig

Le mode d’emploi est inclus dans l’interface (touches [Entrée], [Y], [N], [M], [?], [/], [Echap]). Notez que si vous appuyez sur la touche [Espace] vous faites défiler les choix possibles. N’hésitez pas à utiliser la touche d’aide pour (tenter de) comprendre à quoi sert une option.

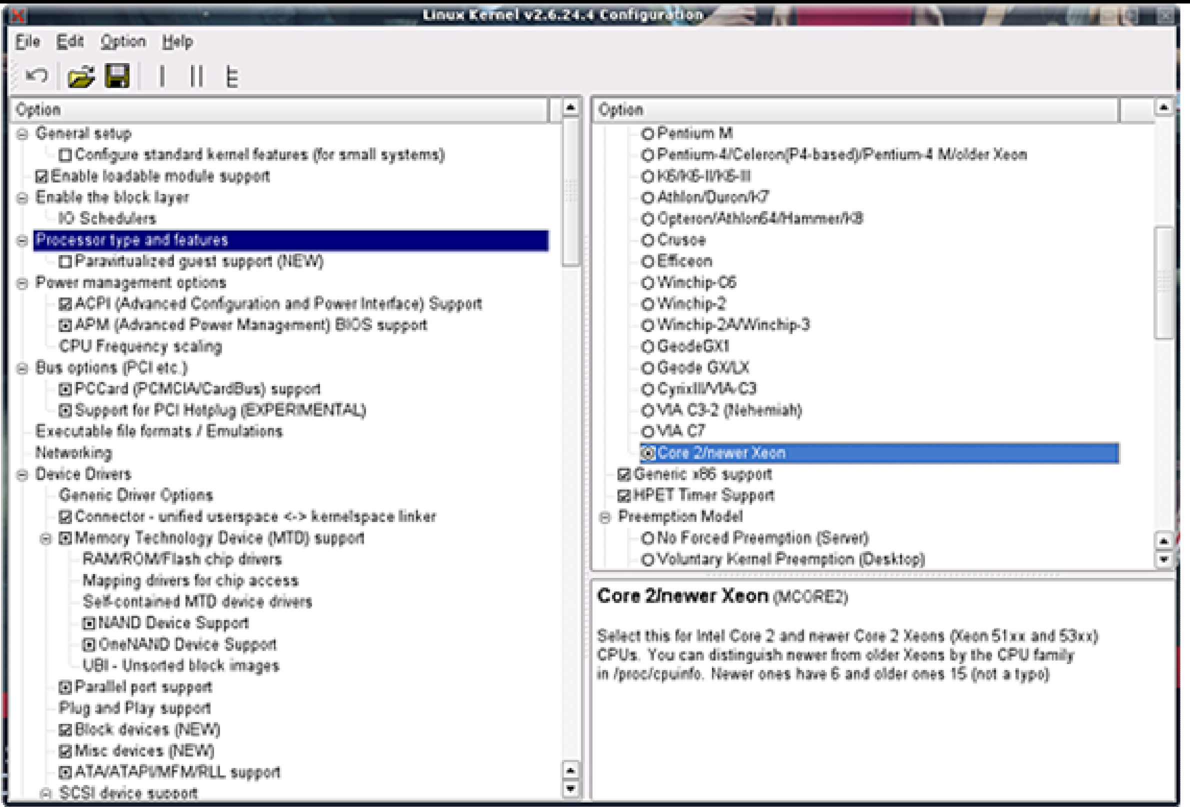
Une fois vos choix effectués, quittez et le fichier .config sera automatiquement généré.

e. make xconfig

Si vous disposez d’un environnement graphique, voici le meilleur choix possible. Vous disposez d’une interface graphique pour configurer votre noyau. La liste des entrées principales est à gauche sous forme d’arborescence. Vous choisissez à droite les options associées. Dans le cadre inférieur vous obtenez de l’aide pour chacune des options. Les menus vous permettent d’effectuer une recherche parmi les valeurs.

Depuis l’interface graphique vous pouvez charger une configuration issue d’un autre fichier de configuration, ou exporter celle-ci vers un fichier de votre choix.

make xconfig



Les options de compilation via xconfig

f. Quelques choix d’optimisation

Les distributions Linux fournissent des noyaux génériques. Certaines proposent des noyaux spécifiques à certains types de processeurs (Intel, AMD, etc.). Dans tous les cas ou presque, des versions 32 bits et 64 bits sont fournies, ces dernières permettant d’exploiter les processeurs Intel ou AMD récents (Athlon64, Prescott, Core2, etc.).

Ces noyaux sont conçus pour fonctionner sur un maximum de machines, processeurs et matériels. Ils sont compilés avec des options standards et la quasi-intégralité du support des divers périphériques. Ils ne sont donc pas optimisés pour certains usages. Vous trouverez très intéressante la possibilité d’adapter le noyau à votre machine ce qui peut vous apporter :

Une parfaite adéquation des optimisations du noyau pour votre matériel.

Un gain certain de performances suivant le type d’usage (serveur, station de travail).

Le support de nouveaux matériels (noyau plus récent).

Un allègement du volume disque occupé par les modules inutiles.



Dans tous les cas, si vous compilez un nouveau noyau, ne désinstallez ou n’écrasez pas l’ancien. Conservez-le : en cas de problème avec votre noyau personnalisé, vous aurez alors la possibilité de revenir en arrière.

Parmi les pistes pour améliorer votre noyau :

Activez les optimisations du noyau pour votre processeur. Dans l'entrée **Processor type and features** sélectionnez par exemple **Core2 / Newer Xeon** si votre machine dispose d'un processeur Core 2 Duo ou supérieur.

Modifiez la valeur du **Preemption Model** en **Preemptible kernel** et cochez **Preempt the Big Kernel Clock** afin de réduire l'effet de latence du noyau (pour le rendre préemptible par les processus).

Modifiez la valeur du **Timer Frequency** à 300 Hz ou 1000 Hz afin de réduire le temps de réponse aux événements.

Dans **CPU Frequency Scaling**, vous pouvez légèrement augmenter la vitesse de démarrage du noyau en passant la valeur de **Default CPUfreq Governor** à **performance**. Ceci ne change rien par la suite où vous pourrez régler après coup la vitesse du processeur.

Supprimez enfin les divers modules dont vous n'aurez jamais besoin. Inutile par exemple de conserver les cartes Token Ring ou ISDN si vous ne les utiliserez jamais.

Sur ce dernier point, faites tout de même attention à ne pas supprimer n'importe quoi. Gardez par exemple à l'esprit que si vous changez de machine, contrairement à Windows vous n'avez pas forcément besoin de réinstaller Linux si le noyau est suffisamment générique (il vous faudra peut-être booter en single pour modifier les modules chargés au démarrage) - l'auteur a migré d'un Pentium 4 d'il y a quatre ans vers un Core 2 Duo (avec changement de carte mère, carte graphique, etc.) sans avoir entièrement réinstallé son système. Le nouveau système a peut-être besoin de modules malheureusement supprimés.

Pensez aussi aux supports et périphériques en hotplug ou en hotswap. Si vous avez par erreur supprimé le pilote ou une option du pilote, vous devrez recompiler le noyau ou redémarrer sur l'ancien, que vous aurez conservé.



Une bonne idée avec les noyaux par défaut des distributions, est de toucher uniquement aux options d'optimisation comme celles vues ci-dessus, en laissant tous les autres choix par défaut.

32 ou 64 bits ?

Quasiment toutes les machines vendues ce jour disposent d'un jeu d'instructions 64 bits, y compris les ordinateurs portables. Pourtant les ordinateurs grand public sont quasiment tous livrés avec des systèmes en 32 bits (Windows XP ou Vista principalement) et ce malgré le fait que des versions 64 bits existent. Le 64 bits a actuellement des avantages et des inconvénients. Ces derniers sont liés à l'historique des logiciels et des systèmes d'exploitation :

Les machines sont livrées avec un OS 32 bits.

Les constructeurs fournissent des pilotes pour ces OS 32 bits.

Les bibliothèques fournies sont compilées en 32 bits.

Les applications liées à ces bibliothèques doivent donc être compilées en 32 bits.

Les constructeurs et éditeurs ne font pas l'effort de fournir des versions 64 bits, la base installée étant trop faible.

Les utilisateurs n'installent pas de version 64 bits, par manque de support de ces dernières.

La boucle s'entretient elle-même, ce qui contraint plus ou moins l'utilisateur à brider son système. Le problème est le même avec les processus en HyperThreading ou multicœurs. L'OS est bien souvent programmé en multithread, mais pas les applications (les jeux sont les bons derniers dans ce cas), ce qui réduit d'autant l'intérêt.

Pourtant une machinant fonctionnant en 64 bits a beaucoup d'avantages :

Un gain important dans la gestion de la mémoire (plus de limitation à 4 Go).

Un gain important de performances sur les applications optimisées.

Une garantie de compatibilité dans l'avenir.

Plus de "big crunch" (passage à zéro des compteurs de temps sous Unix ayant pour effet de repasser toutes les dates au 13 décembre 1901) Unix en janvier 2038 !

À titre de comparaison, le transcodage d'un DVD en DIVX via dvdrip (options : une passe, désentrelacement intelligent, grande taille redimentionnement HQ, mp3 192 kbits) tourne à une vitesse de 27 à 30 images par seconde sur une openSUSE 10.3 en 32 bits (Core 2 Duo e6750). Sur la même machine, la vitesse est de 40-42 images par seconde (sur une installation optimisée pour travailler en 64 bits). Soit un gain moyen de 25 à 30%. La même chose a été constatée avec un calcul super_pi optimisé. Le gain peut être réel, même s'il n'est pas significatif dans un environnement bureautique.

Comme les sources du noyau sont fournies et que la plupart des pilotes y sont inclus, soit fournis sous forme de sources, tout matériel supporté en 32 bits par Linux l'est en principe en 64 bits. Certains modules propriétaires (ati, nvidia) sont fournis en 64 bits. Les distributions 64 bits fournissent le nécessaire (bibliothèques) pour faire fonctionner les applications 32 bits nativement.

Aussi, si vous en avez la possibilité, testez une version 64 bits de Linux.



Si vous recompilez un noyau avec les extensions 64 bits depuis une distribution 32 bits, seul le noyau fonctionnera en 64 bits. Il faudrait alors installer ensuite toutes les bibliothèques et

tous les outils en 64 bits, ce qui est complexe et long. Le mieux dans ce cas est de réinstaller le système.

4. Compilation

Maintenant que le noyau est configuré, vous pouvez lancer la compilation. Cette étape est la plus longue. Le noyau contient des millions de lignes en langage C et en assembleur. Suivant vos options (et surtout les modules inclus), la charge de votre machine et la vitesse de votre processeur, cette étape peut prendre de quelques minutes à quelques heures ! Dans tous les cas, vous avez le temps pour faire autre chose.

```
# make

scripts/kconfig/conf -s arch/x86/Kconfig

  CHK     include/linux/version.h

  UPD     include/linux/version.h

  CHK     include/linux/utsrelease.h

  UPD     include/linux/utsrelease.h

  SYMLINK include/asm -> include/asm-x86

  CC      arch/x86/kernel/asm-offsets.s

  GEN     include/asm-x86/asm-offsets.h

  CALL    scripts/checksyscalls.sh

  HOSTCC  scripts/genksyms/genksyms.o

  SHIPPED scripts/genksyms/lex.c

  SHIPPED scripts/genksyms/parse.h

  SHIPPED scripts/genksyms/keywords.c

  HOSTCC  scripts/genksyms/lex.o

  SHIPPED scripts/genksyms/parse.c

  HOSTCC  scripts/genksyms/parse.o

  HOSTLD  scripts/genksyms/genksyms

  CC      scripts/mod/empty.o
```

```
  HOSTCC  scripts/mod/mk_elfconfig
  MKELF   scripts/mod/elfconfig.h
  HOSTCC  scripts/mod/file2alias.o
  HOSTCC  scripts/mod/modpost.o
  HOSTCC  scripts/mod/sumversion.o
  HOSTLD  scripts/mod/modpost
  HOSTCC  scripts/kallsyms
  HOSTCC  scripts/conmakehash
  HOSTCC  scripts/bin2c
  CC      init/main.o
  CHK     include/linux/compile.h
  UPD     include/linux/compile.h
  CC      init/version.o
  CC      init/do_mounts.o
…
  BUILD   arch/x86/boot/bzImage
Root device is (8, 6)
Setup is 11012 bytes (padded to 11264 bytes).
System is 1567 kB
Kernel: arch/x86/boot/bzImage is ready  (#1)
  Building modules, stage 2.
  MODPOST 1950 modules
…
  LD [M]  sound/usb/snd-usb-lib.ko
  CC      sound/usb/usx2y/snd-usb-usx2y.mod.o
```

LD [M] sound/usb/usx2y/snd-usb-usx2y.ko

La première colonne est l'action effectuée sur le fichier représenté par la deuxième colonne. CC indique une compilation, LD l'édition des liens, un [M] l'action sur un module, etc.

5. Installation

Si la compilation s'est effectuée sans erreurs, il vous reste deux actions à effectuer. Tout d'abord installez les modules. La commande suivante les installe dans **/lib/module/<version_noyau>** et crée le fichier des dépendances associés.

```
# make modules_install

  INSTALL arch/x86/crypto/aes-i586.ko

  INSTALL arch/x86/crypto/twofish-i586.ko

  INSTALL arch/x86/kernel/apm.ko

  INSTALL arch/x86/kernel/cpu/cpufreq/acpi-cpufreq.ko

  INSTALL arch/x86/kernel/cpu/cpufreq/cpufreq-nforce2.ko

  INSTALL arch/x86/kernel/cpu/cpufreq/e_powersaver.ko

  INSTALL arch/x86/kernel/cpu/cpufreq/gx-suspmod.ko

  INSTALL arch/x86/kernel/cpu/cpufreq/longhaul.ko

  INSTALL arch/x86/kernel/cpu/cpufreq/longrun.ko

  INSTALL arch/x86/kernel/cpu/cpufreq/p4-clockmod.ko

  INSTALL arch/x86/kernel/cpu/cpufreq/powernow-k6.ko

…

  INSTALL sound/usb/snd-usb-lib.ko

  INSTALL sound/usb/usx2y/snd-usb-usx2y.ko

  DEPMOD  2.6.24.4-default
```

Cette seconde commande recopie le noyau et le nécessaire associé dans /boot. Selon les distributions, elle va aussi créer l'initrd associé, et modifier la configuration du chargeur de démarrage GRUB.

```
# make install
```

sh /usr/src/linux-2.6.24.4/arch/x86/boot/install.sh 2.6.24.4-default

arch/x86/boot/bzImage System.map "/boot"

Kernel image: /boot/vmlinuz-2.6.24.4-default

Initrd image: /boot/initrd-2.6.24.4-default

Root device: /dev/disk/by-id/scsi-SATA_ST380011A_5JVTH798-part6

(/dev/sda6) (mounted on / as ext3)

Resume device: /dev/sda5

Kernel Modules: processor thermal scsi_mod libata sata_sil pata_atiixp

fan jbd mbcache ext3 edd sd_mod usbcore ohci-hcd uhci-hcd ehci-hcd

ff-memless hid usbhid

Features: block usb resume.userspace resume.kernel

Bootsplash: SuSE (1280x1024)

36359 blocks

Respectez l’ordre précédent. Si vous installez le noyau avant ses modules, l’initrd ne pourra pas être construit car les modules devant y être présents ne sont pas encore installés. Il se peut que l’initrd ne soit pas généré par défaut, auquel cas vous devrez après l’installation relancer cette commande avec les paramètres correspondant au noyau installé.

ls -l /boot/*2.6.24*

-rw-r--r-- 1 root root 7877336 avr 8 19:08 /boot/initrd-2.6.24.4-

default

-rw-r--r-- 1 root root 872762 avr 8 19:06 /boot/System.map-2.6.24.4-

default

-rw-r--r-- 1 root root 1615232 avr 8 19:07 /boot/vmlinuz-2.6.24.4-

default

Vérifiez si la configuration du chargeur de démarrage a été modifiée.

grep 2.6.24 /boot/grub/menu.lst

title openSUSE 10.3 - 2.6.24.4

kernel /boot/vmlinuz-2.6.24.4-default root=/dev/disk/by-id/scsi-

SATA_ST380011A_5JVTH798-part6 vga=0x31a resume=/dev/sda5 splash=silent

showopts

initrd /boot/initrd-2.6.24.4-default

title Failsafe -- openSUSE 10.3 - 2.6.24.4

kernel /boot/vmlinuz-2.6.24.4-default root=/dev/disk/by-id/scsi-

SATA_ST380011A_5JVTH798-part6 vga=normal showopts ide=nodma apm=off

acpi=off noresume nosmp noapic maxcpus=0 edd=off 3

initrd /boot/initrd-2.6.24.4-default



Vous n’avez pas à réinstaller GRUB quand vous modifiez son fichier de configuration.

Si vous utilisez LILO, vous devez, outre modifier **/etc/lilo.conf**, le réinstaller avec **/sbin/lilo**.

6. Test

Si toutes les étapes précédentes ont fonctionné, il n’y a plus qu’à redémarrer votre ordinateur et à sélectionner votre nouveau noyau au chargement. Si le boot se termine correctement (accès à la console ou environnement graphique), ouvrez une console et vérifiez la version de votre système.

$ uname -a

Linux slyserver 2.6.24.4-default #1 SMP PREEMPT Tue Apr 8 19:06:10

CEST 2008 x86_64 intel x86_64 GNU/Linux

7. Autres options

Après une compilation, les divers fichiers intermédiaires (fichiers objets) prennent énormément de place. Pour les supprimer, utilisez la commande suivante : **# make clean**

Si votre distribution est à base de rpm, vous pouvez créer les packages du noyau (sources, headers, noyau) avec la commande suivante : **# make rpm**

THEME 106 Séquence de démarrage, d’arrêt et niveaux

106.1 Démarrer le système (3)

Processus de démarrage

1. Le BIOS

a. Rôle

Le **BIOS** (**Basic Input Output System**) est l’interface logicielle entre le matériel et le logiciel à un niveau très basique. Il fournit l’ensemble des instructions de base utilisées par le système d’exploitation. Il fournit le niveau d’interface le plus bas aux pilotes et périphériques.

Le BIOS est présent sur une mémoire **EEPROM** (**Electrical Erasable Programmable Read-Only Memory**) de l’ordinateur. Quand l’ordinateur est électriquement allumé, ou lors d’un reset, un signal appelé **powergood** est envoyé au microprocesseur. Celui-ci déclenche alors l’exécution du BIOS.

Le BIOS effectue un auto-test de l’allumage (POST) puis recherche les périphériques, notamment ceux utilisés pour démarrer. Les informations sur le matériel sont stockées de manière permanente dans une petite mémoire CMOS alimentée par une batterie. À la fin du processus, le périphérique de démarrage est sélectionné.

Le BIOS lit et exécute le premier secteur physique du média de démarrage. Il s’agit généralement des 512 premiers octets du premier disque dur (le MBR) ou de la partition active (le PBR), comme le chapitre Les disques et les systèmes de fichiers vous l’a décrit.

b. Réglages basiques

Chaque BIOS est différent selon les constructeurs de cartes mères et les éditeurs (AMIBios, Phoenix, Award, etc.). Cependant de nombreux réglages sont identiques ou en tout cas se ressemblent en passant de l’un à l’autre.

C’est par le BIOS que s’effectue la détection des disques durs et le choix du support de démarrage. Linux supporte l’IDE, le SATA et le SCSI. Il se peut cependant qu’en SATA votre chipset ne soit pas reconnu. Dans ce cas, la plupart des BIOS permettent de passer le contrôleur SATA en mode d’émulation IDE. Linux les reconnaîtra comme tels. Cependant vous ne perdrez rien à tester une première installation avec le support natif des disques en SATA activé.

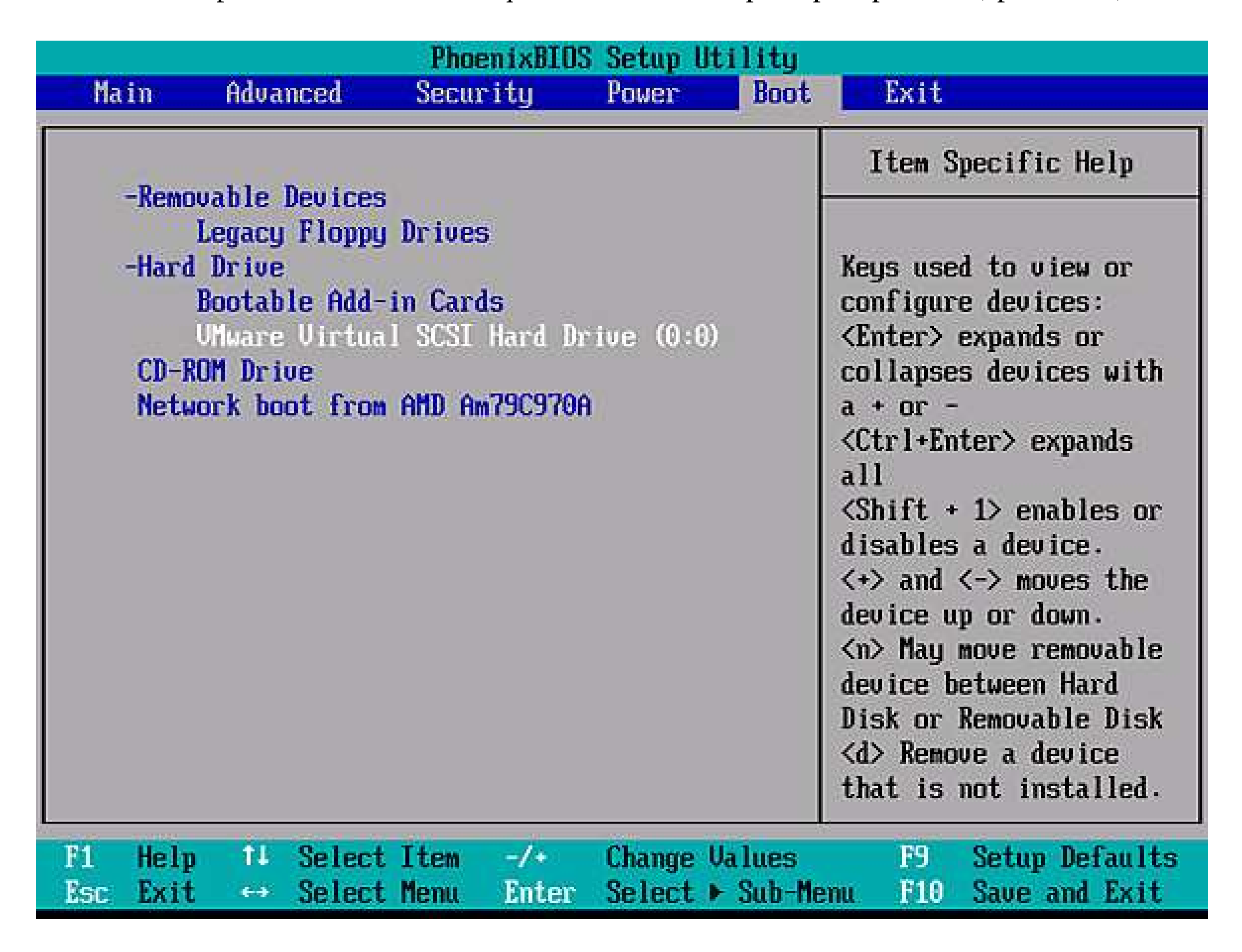
En principe, Linux gère bien le support des chipsets SATA compatibles **AHCI** (**Advanded Host Controller Interface**), un standard aux spécifications publiques. Activez ce choix dans le BIOS s’il vous est proposé, il apparaît sous ce nom, et parfois en **native**. Si rien ne

fonctionne, tentez le mode **combined**, puis **legacy IDE**. Vous trouverez de l'aide sur le SATA sur le site suivant : http://linux-ata.org/faq.html

Pour démarrer l'installation de Linux depuis un CD-Rom ou un DVD-Rom vous devez modifier l'ordre de démarrage dans la séquence de boot pour démarrer en premier sur le lecteur CD ou DVD.

Si votre clavier est de type USB, ou sans fil mais avec un adaptateur sans fil USB, vous devez activer le **USB legacy support** (cette fonction s'appelle parfois **USB DOS function ou USB keyboard enable**). Il permet d'activer au boot le support des claviers mais aussi des supports de stockage USB (clés, disques durs, cartes mémoire). Ceci n'empêche pas la prise en charge de l'USB par le système : une fois l'OS démarré, ce sont les pilotes USB du noyau et des modules qui prennent en charge l'USB.

Vous n'avez pas, en principe, à toucher aux autres réglages. Évitez notamment de jouer à l'apprenti sorcier en modifiant les réglages avancés du chipset et autres ressources dont vous ne comprenez pas l'utilité. Vous pouvez cependant dans un souci d'économie de ressources désactiver les ports de la carte mère que vous n'utilisez pas : port parallèle, port série, etc.



Écran du BIOS Phoenix modifiant l'ordre de boot



L'overclocking nécessite du matériel adapté : processeur, carte mère, mémoire et alimentation doivent être de haute qualité et le PC bien ventilé. L'overclocking est à la base de nombreuses sources d'instabilité et de plantages, tant sous Windows que sous Linux. La mémoire notamment est soumise à rude épreuve. Elle est la principale cause d'instabilité. Même sans overclocking, il n'est pas vain d'investir dans des composants de qualité.

2. Le chargeur de démarrage

Le BIOS active le chargeur de programme initial (**Initial Program Loader**, IPL) à partir des premiers 512 octets du support de démarrage. Sur Linux, le chargeur est décomposé en deux parties. Le chargeur initial des 512 octets ne contient pas assez de code pour proposer des menus et lancer le chargement d'un système d'exploitation. Il charge la seconde phase, basée sur un fichier de configuration.

La seconde phase fournit une interface pour lancer un système d'exploitation parmi un choix donné. Vous pouvez en profiter pour passer des paramètres au noyau Linux et au processus init.

Le BIOS n'intervient qu'au démarrage de la machine, à l'utilisation du chargeur de démarrage et aux toutes premières étapes du chargement du noyau. Ensuite, il devient inutile. Le noyau dispose de ses propres fonctions de détection bien qu'il s'appuie sur la configuration du BIOS. En effet ce dernier, sur plate-forme Intel, s'exécute en mode réel et Linux en mode protégé.

3. GRUB

a. Configuration

Le chargeur par défaut sur la plupart des distributions Linux s'appelle **GRUB** (**Grand Unified Bootloader**). Il est hautement paramétrable, notamment en acceptant une protection par mot de passe crypté, un interpréteur de commandes ou encore des graphiques. Il est basé sur un fichier texte de configuration et il n'y a pas besoin de réinstaller GRUB à chaque modification.

Voici un exemple de configuration partant du principe que la première partition du premier disque est /boot, et que la seconde contient une installation de Windows.

```
timeout=10

default=0

title Red Hat

   root (hd0,0)

   kernel /vmlinuz-2.6.12-15 ro root=LABEL=/

   initrd /initrd-2.6.12-15.img
```

title Windows XP

rootnoverify (hd0,1)

chainloader +1

Voici la syntaxe générale d'un fichier GRUB :

Paramètre GRUB	Signification
timeout	Nombre de secondes avant le démarrage par défaut.
default n	Démarrage par défaut (0=1er titre, 1=2ème titre, etc.).
gfxmenu	Chemin vers un menu graphique.
title xxxx	Début d'une section, entrée du menu de GRUB.
root(hdx,y)	Tous les accès fichiers spécifiés dessous le seront à partir de cette partition (cf. signification plus bas). Ici hd0,0 représente la première partition du premier disque détecté par le BIOS. C'est la partition /boot.
kernel	Le nom de l'image du noyau Linux, suivi de ses paramètres. Le / n'indique pas la racine du système de fichiers mais celle de (hd0,0), donc /boot/vmlinuz...
initrd	Initial ramdisk. Le noyau va charger ce fichier comme disque en mémoire pour y trouver une configuration et des pilotes initiaux.
rootnoverify	La racine spécifiée, à ne pas monter par GRUB (il ne supporte pas NTFS).
chainloader +1	Démarrer le premier secteur de la racine spécifiée ci-dessus.

Voici la signification des noms de périphériques sous GRUB.

(fd0) : premier lecteur de disquettes détecté par le BIOS (/dev/fd0 sous Linux).

(hd0,0) : première partition sur le premier disque dur détecté par le BIOS que ce soit IDE ou SCSI (/dev/hda1 ou /dev/sda1 suivant le cas).

(hd1,4) : cinquième partition sur le second disque dur détecté par le BIOS (/dev/hdb5 ou /dev/sda5).

b. Installation

La configuration de **GRUB** réside dans **/etc/grub.conf** ou **/boot/grub/menu.lst** (le premier est un lien sur l'autre). GRUB peut s'installer sur un **MBR** (**Master Boot Record**, les 512 premiers octets d'un disque) ou un **PBR** (**Partition Boot Record**, les 512 premiers octets d'une partition).

Pour installer ou réinstaller GRUB en cas de MBR corrompu, par exemple sur **/dev/sda** utilisez la commande **grub-install** :

/sbin/grub-install /dev/sda

c. Démarrage et édition

Au démarrage de GRUB, un menu s'affiche. Il peut être graphique ou textuel, selon la configuration. Vous devez choisir une image de démarrage avec les flèches de direction parmi celles proposées. En appuyant sur la touche [Entrée] vous démarrez l'image sélectionnée.

Vous pouvez éditer les menus directement pour modifier par exemple les paramètres passés au noyau Linux ou init. Dans ce cas, sélectionnez une entrée de menu et appuyez sur la touche **e** (edit). Ici, toutes les lignes de la section sont affichées. Vous pouvez appuyer sur :

e : pour éditer la ligne (la compléter) ;

d : pour supprimer la ligne ;

o : pour ajouter une ligne ;

b : pour démarrer l'image (booter).

Par exemple, pour démarrer en mode urgence (emergency) :

Allez sur la ligne Linux ou Red Hat et appuyez sur **e**.

Allez sur la ligne kernel et appuyez sur **e**.

À la fin de la ligne rajoutez 1 ou Single et appuyez sur [Entrée].

Appuyez sur **b**.

Vous pouvez aussi accéder à un interpréteur de commandes en appuyant sur [Echap]. Attention seules les commandes GRUB sont reconnues.

4. Initialisation du noyau

Au chargement du noyau une multitude d'informations défile sur l'écran. Vous ne pouvez pas figer ces informations à ce moment-là. Par contre juste après le passage à l'étape suivante (init) toutes les traces du noyau sont placées dans le fichier **/var/log/dmesg**.

Le matériel est détecté et initialisé.

initrd est chargé, les modules présents éventuellement chargés.

Le noyau monte le système de fichiers racine en lecture seule.

Il crée la première console.

Le premier processus est lancé (normalement init).





Démarrage électrique

BIOS

MBR / PBR
LILO | GRUB

Boot sur autre partition (chainloader)

Windows XP, Vista, etc.

Noyau Linux
initrd

init

Schéma de la séquence de démarrage

Il existe d'autres chargeurs dont un appelé **LILO (Linux Loader)**. Celui-ci a été jusqu'à peu le chargeur le plus utilisé. Ces dernières années, il a été cependant presque totalement remplacé par GRUB à cause de nombreuses limitations.

TP : Démarrage de Linux, services, noyau et périphériques - GRUB et le processus de boot

106.2 Changer les niveaux, arrêter ou rebooter

init

1. Rôle

Le programme init, premier démarré et dernier stoppé au sein du système, est celui qui lance toutes les autres tâches. Le rôle initial de init est de démarrer et d'arrêter tous les services. C'est init qui va exécuter les diverses tâches initiales nécessaires au bon fonctionnement de Linux via l'exécution de plusieurs commandes et scripts.

Une fois le système démarré et les services lancés, init reste actif pour gérer les changements d'état des processus qu'il contrôle et des niveaux d'exécution.

Le programme init n'est pas toujours le même d'une distribution à une autre. Sur la plupart des distributions professionnelles et/ou majeures (Mandriva, Red Hat/Fedora, openSUSE, Debian, etc.) le principe est globalement le même : init de type System V (basé sur la notion de niveaux d'exécution). La distribution Ubuntu utilise Upstart qui gère les priorités, les événements et dépendances entre les services, mais qui reste calquée sur le même principe. La distribution Slackware utilise un autre principe issu du fonctionnement de BSD.

Le processus **init** est le père de tous les processus. Il a toujours le PID 1. Sa configuration est présente dans le fichier **/etc/inittab**. Si ce fichier est corrompu et inutilisable, il faudra démarrer en mode single (S, s, 1, Single) et le réparer, ou au pire démarrer depuis un support externe ou un disque de secours. C'est un fichier central du système d'exploitation.

2. Niveaux d'exécution

Un niveau d'exécution, ou runlevel, est un état dans lequel se trouve Unix/Linux. Cet état est contrôlé par init. Chaque état dispose de sa propre configuration (soit par inittab, soit par des scripts appelés initscripts). Un niveau d'exécution peut par exemple être utilisé pour lancer Unix en mono-utilisateur, en multi-utilisateurs, avec ou sans réseau, avec ou sans mode graphique. Tous les niveaux sont personnalisables par l'administrateur. Ces niveaux sont généralement définis comme ceci par convention sur les distributions Red Hat/Fedora, Mandriva, openSUSE et associées :

Niveau	Effet
0	Halt : stoppe le système d'exploitation, éteint la machine.
1	Mode mono-utilisateur utilisé pour la maintenance, mode console.
2	Multi-utilisateur, sans réseau, console.
3	Multi-utilisateur, avec réseau, console.
4	Idem que le 3, laissé à la convenance de l'administrateur.
5	Multi-utilisateur, avec réseau, avec environnement graphique X Window.
6	Reboot : redémarrage de la machine.
S,s	Single user mode, le mode le plus bas en cas de soucis.

Les niveaux 7 à 9 sont parfaitement valides mais pas utilisés par défaut. Le niveau d'exécution par défaut est positionné dans **/etc/inittab** sur la ligne initdefault.

id:5:initdefault:

Remplacez 5 par le niveau souhaité au démarrage.



Un gag, posé lors de certains exercices éliminatoires de certifications, consiste à créer une situation de panne où la valeur par défaut est positionnée à 0 ou 6. Dans le premier cas la machine s'éteint dès l'exécution de init, dans l'autre elle redémarre en boucle…

La distribution Debian (et les distributions qui en dérivent) considère aussi les niveaux 2 à 5 comme multi-utilisateur mais n'établit pas de différences entre ces niveaux, et démarre par défaut au niveau 2 où tout est lancé, y compris éventuellement l'interface graphique.

Comme chaque niveau peut être totalement modifié et reconfiguré, il est possible de tout redéfinir et donc de faire en sorte qu'une Debian démarre comme une Red Hat, ou qu'une Red Hat démarre comme une Debian. Pour des raisons de conformité et de support, considérez qu'il est important de rester conforme au « standard » de la distribution que vous utilisez.

3. /etc/inittab

Le comportement du processus init et des runlevels est défini dans le fichier **/etc/inittab**. La syntaxe d'une ligne est la suivante :

Id:[niveaux]:action:commande

Champ	Description
Id	Identifiant de ligne sur quatre caractères, juste indicatif (sous Linux avec getty/mingetty : numéro de terminal).
Niveaux	Indique si la commande doit être prise en compte pour le niveau demandé, c'est la liste des niveaux sans séparateur.
Action	Type d'action à effectuer selon les circonstances pour cette ligne.
Commande	La commande à exécuter avec ses paramètres et les redirections.

L'action est très importante car elle définit les activités de init lors du démarrage et du changement de niveau. Voici les principales :

Action	Signification
initdefault	Définit le niveau par défaut lors du boot et du lancement d'init.
sysinit	Exécuté une seule et unique fois lors du démarrage du système.
boot	Idem mais après sysinit.
bootwait	Idem, mais init attend la fin de l'exécution de la commande avant de continuer à dérouler le fichier inittab.

Action	Signification
off	La ligne est ignorée.
once	La commande est exécutée à chaque changement de niveau pour les niveaux spécifiés.
wait	Idem, mais init attend la fin de l'exécution avant de continuer.
respawn	La commande est lancée pour les niveaux concernés. Si le processus se termine, il est automatiquement relancé. C'est le cas pour les terminaux si un utilisateur s'en déconnecte.
powerwait	La commande est lancée si le serveur passe sur alimentation de secours (UPS).
powerfail	Idem, mais sans attente de la fin d'exécution de la commande.
powerokwait	La commande est lancée lorsque le courant est rétabli.
powerfailnow	Commande de dernier recours lorsque l'alimentation de secours est presque vide.
ctrlaltdel	Init reçoit un signal SIGINT issu d'une séquence [Alt][Ctrl][Suppr].

Voici un exemple issu d'une installation openSUSE 10.3 :

```
# Niveau d'exécution à 5 (multiuser graphique)

id:5:initdefault:



# Premier script execute au démarrage

si::bootwait:/etc/init.d/boot



# Gestion des services par niveau d'exécution

l0:0:wait:/etc/init.d/rc 0

l1:1:wait:/etc/init.d/rc 1

l2:2:wait:/etc/init.d/rc 2

l3:3:wait:/etc/init.d/rc 3

l4:4:wait:/etc/init.d/rc 4

l5:5:wait:/etc/init.d/rc 5

l6:6:wait:/etc/init.d/rc 6
```

Cas du mode single, console de secours pour root

ls:S:wait:/etc/init.d/rc S

~~:S:respawn:/sbin/sulogin

Action sur Alt+Ctrl+Del

ca::ctrlaltdel:/sbin/shutdown -r -t 4 now

Que faire en cas de coupure de courant

pf::powerwait:/etc/init.d/powerfail start

pn::powerfailnow:/etc/init.d/powerfail now

po::powerokwait:/etc/init.d/powerfail stop

Lancement des consoles virtuelles Alt+Fx

1:2345:respawn:/sbin/mingetty --noclear tty1

2:2345:respawn:/sbin/mingetty tty2

3:2345:respawn:/sbin/mingetty tty3

4:2345:respawn:/sbin/mingetty tty4

5:2345:respawn:/sbin/mingetty tty5

6:2345:respawn:/sbin/mingetty tty6

4. Changement de niveau

Vous pouvez changer de niveau à la volée après le démarrage de la machine avec la commande **/sbin/init** ou **/sbin/telinit**, cette dernière étant un simple lien symbolique vers init. La commande suivante passe au niveau 5.

telinit 5

Les valeurs **q**, **u** ou **-t** peuvent être précisées :

Q ou **q** : init relit le fichier **/etc/inittab**, s'il a été modifié, en corrigeant ses tables internes.

U ou **u** : init se relance sans relire inittab et sans changer de niveau. Si des services ont été rajoutés ou supprimés du niveau en cours, init prend en compte la modification.

-t : quand init a terminé l'arrêt des services (ou plutôt quand le script rc l'a fait, voir un peu plus loin), init envoie le signal SIGTERM à tous les processus restants, leur demandant de se terminer proprement, attend le nombre de secondes spécifié (5 par défaut), puis envoie SIGKILL.

Le niveau d'exécution actuel est visible avec la commande **/sbin/runlevel**. La première valeur retournée est le niveau précédent le niveau actuel. Un N signifie qu'il n'y a pas de précédent niveau. La seconde valeur est le niveau actuel.

runlevel

N 5

5. Paramétrage système de base

Quel que soit le niveau d'exécution précisé par défaut, init lance toujours la commande associée aux actions sysinit, bootwait ou boot de **/etc/inittab** lors du démarrage du système, l'action sysinit étant la première.

Sous Red Hat : **si::sysinit:/etc/rc.d/rc.sysinit**

Sous openSUSE : **si::bootwait:/etc/init.d/boot**

Sous Debian : **si::sysinit:/etc/init.d/rcS**

Sous Red Hat, c'est un seul script monolithique qui s'occupe de toute la configuration de base. Sous Debian, le script appelle tous les scripts du niveau S (single). Sous openSUSE le script met en place le strict nécessaire puis exécute le contenu de **/etc/rc.d/boot.d** qui établit le reste de la configuration de base.

Dans tous les cas, les tâches suivantes sont exécutées à peu près dans cet ordre :

Configuration des paramètres du noyau présents dans **/etc/sysctl.conf** (ex : IP Forwarding).

Mise en place des fichiers périphériques (/dev via udev par exemple).

Configuration de l'horloge du système.

Chargement des tables de caractères du clavier.

Activation des partitions d'échange SWAP.

Définition du nom d'hôte.

Contrôle et montage du système de fichiers racine (en lecture-écriture cette fois).

Ajout des périphériques RAID et/ou LVM. Ceci peut déjà être mis en place lors du chargement de inittab.

Activation des quotas de disque.

Contrôle et montage des autres systèmes de fichiers.

Nettoyage des verrous (stale locks) et des fichiers PID en cas d'arrêt brusque.

Il est possible avec certaines distributions de passer en mode interactif. Au début du boot, après le lancement d'init, il peut vous être demandé de taper sur la lettre i puis de répondre par oui ou par non aux différentes actions.

6. Niveaux d'exécution System V

a. rc

Le script **/etc/init.d/rc** prend comme paramètre le niveau d'exécution par défaut selon la ligne **initdefault** de **/etc/inittab** ou celui spécifié lors de l'appel manuel des commandes **init** ou **telinit**. Le script rc initialise le niveau d'exécution voulu et est responsable du démarrage et de l'arrêt des services associés quand le niveau d'exécution change.

l1:1:wait:/etc/init.d/rc 1

l2:2:wait:/etc/init.d/rc 2

l3:3:wait:/etc/init.d/rc 3

l4:4:wait:/etc/init.d/rc 4

l5:5:wait:/etc/init.d/rc 5

l6:6:wait:/etc/init.d/rc 6

Les services sont analysés à chaque niveau d'exécution. Lors du passage d'un niveau à un autre, et quel que soit l'ordre (du 2 au 5, du 5 au 3, etc.) le script rc compare les services qui doivent être arrêtés ou démarrés entre l'ancien et le nouveau niveau. Si un service est commun aux deux niveaux, il est maintenu. Si un nouveau service doit être lancé dans le nouveau niveau, il le lance. Si un service doit être arrêté car il est absent du nouveau niveau, il l'arrête.



Ce fonctionnement, standard à toutes les distributions Linux de type System V, n'est pas commun à tous les Unix. HP-UX (un Unix de HP) considère qu'il doit y avoir une progression constante dans les niveaux, passant successivement du 1 au 3 (1 puis 2 puis 3) et chargeant successivement les services. À l'arrêt il redescend jusqu'au niveau 0 en terminant

successivement les services. La différence est de taille : il ne compare pas les niveaux et n'effectue pas d'arrêt/relance entre chaque niveau…

7. Gestion des niveaux et des services

a. Services dans init.d

Le niveau d'exécution définit les services à démarrer pour ce niveau. C'est le script rc qui charge les services. Les services sont contrôlés (démarrage, arrêt, relance, status, etc.) à l'aide de scripts présents dans **/etc/init.d**.

```
# cd /etc/init.d

# ls -l

-rwxr-xr-x  1 root root  1128 aou  9  2004 acpid

-rwxr-xr-x  1 root root   834 sep 28  2004 anacron

-rwxr-xr-x  1 root root  1429 jun 22  2004 apmd

-rwxr-xr-x  1 root root  1176 avr 14  2006 atd

-rwxr-xr-x  1 root root  2781 mar  5  2007 auditd

-rwxr-xr-x  1 root root 17058 sep  5  2007 autofs

-rwxr-xr-x  1 root root  1368 fev  2  2007 bluetooth

-rwxr-xr-x  1 root root  1355 mai  2  2006 cpuspeed

-rwxr-xr-x  1 root root  1904 jui 16  2007 crond

-rwxr-xr-x  1 root root  2312 oct 30 13:46 cups

…
```

Pour chaque niveau d'exécution **n**, il existe un répertoire rc**n**.d qui contient des liens symboliques (raccourcis) vers les services présents dans **/etc/init.d** à lancer ou arrêter. Ce répertoire peut être à différents endroits selon la distribution :

Red Hat : **/etc/rc.d/rcn.d** avec des liens sur **/etc/rcn.d**

openSUSE : **/etc/init.d/rcn.d** sachant que **/etc/rc.d** pointe sur **/etc/init.d**

Debian : **/etc/rcn.d**

Le préfixe du nom de chaque lien définit son ordre de lancement ou son ordre d'arrêt. Le nom est sous la forme suivante :

[SK]nnservice

S : start.

K : kill (stop).

nn : ordre numérique de démarrage ou d’arrêt. (00=premier, 99=dernier).

service : nom du service.

Par exemple le lien S10network indique que le service network, responsable de la mise en place du réseau, sera démarré en ordre 10, après les S01, S05, etc. mais avant les S11, S15, S20, etc.

ls -1 S*

S00microcode_ctl

S01sysstat

S02lvm2-monitor

S05kudzu

S06cpuspeed

S08iptables

S09isdn

S09pcmcia

S10network

S12syslog

S13irqbalance

S13portmap

S14nfslock

S15mdmonitor

S18rpcidmapd

…

Quand rc est exécuté, il va tout d'abord lister tous les liens commençant par K* à l'aide d'une boucle for. Puis il fait la même chose pour S*, cette fois en lançant les services. Voici un bout du fichier rc pour mieux comprendre la séquence de démarrage :

```
# test existence de /etc/rcn.d

if [ -d /etc/rc${level}.d ]

then

   # Liste tous les scripts commençant par S dans ce répertoire

   for i in /etc/rc${level].d/S*

   do

      # le script existe et n'est pas vide : on l'exécute

      if [ -s ${i} ]

          then

             sh ${i} start

      fi

   done

fi
```

b. Contrôle manuel des services

Via le script

Les services peuvent être lancés dans tous les cas individuellement, ou à l'aide d'outils selon la distribution. La première méthode est la seule par défaut sous Debian. Chaque service présent dans /etc/init.d accepte au moins deux paramètres :

start : le service démarre.

stop : le service s'arrête.

Si vous souhaitez démarrer et arrêter le service sshd (serveur ssh) à la main :

```
# /etc/init.d/sshd start

Starting SSH daemon                                    done

# /etc/init.d/sshd stop
```

Shutting down SSH daemon done

Certains services peuvent accepter d'autres paramètres :

/etc/init.d/sshd

Usage: /etc/init.d/sshd {start|stop|status|try-

restart|restart|force-reload|reload|probe}

status : fournit l'état du service (démarré ou non). Selon les services des informations supplémentaires peuvent être fournies.

probe : indique s'il y a nécessité de recharger la configuration, si des fichiers de configuration ont par exemple été modifiés.

reload / forcereload : indique au service de relire sa configuration (via un signal 1).

restart : arrête et relance le service, quelle que soit l'issue de l'arrêt.

try-restart : arrête et relance le service seulement si l'arrêt a bien eu lieu.



La distribution openSUSE crée des liens symboliques **rc<service>** permettant de saisir **rcsshd** par exemple pour le contrôle manuel des services.

Via la commande service

La commande **service** est disponible sous Red Hat et openSUSE. Elle permet simplement de se passer du chemin vers le script de lancement du service et d'utiliser simplement son nom :

service sshd stop

Shutting down SSH daemon done

service sshd start

Starting SSH daemon done

Pour contrôler la configuration des services de System V lancés par init, il n'est pas conseillé de tout faire à la main mais plutôt d'utiliser les outils du système concerné quand ils existent, en mode texte ou graphique.

c. Modification des niveaux d'exécution

Red Hat et openSUSE

Sous Red Hat/Fedora et openSUSE la commande **chkconfig** permet d'ajouter, de supprimer, d'activer ou de désactiver des scripts, par niveau d'exécution. Cette commande est très pratique pour configurer les services parce qu'elle sait gérer tant les services System V que les services xinetd.

chkconfig [option] [service]



Bien que la syntaxe soit identique dans les deux distributions, chkconfig ne fonctionne pas de la même manière. Sous Red Hat une ligne spéciale est insérée en début de script qui indique à chkconfig les paramètres par défaut (runlevels, positions de démarrage et d'arrêt). Sous openSUSE, chkconfig est un frontend pour la commande **insserv**. Cette dernière exploite aussi l'en-tête des scripts, mais de manière plus complexe (elle gère l'ordonnancement et le parallélisme par exemple).

Voici le début du script de lancement de service sshd sous Red Hat. Le script démarre et s'arrête dans les niveaux 2, 3, 4 et 5. Il démarre en position 55 (S55sshd) et s'arrête en position 25 (K25sshd).

chkconfig: 2345 55 25

description: OpenSSH server daemon

Voici la même chose sous openSUSE. chkconfig et insserv gèrent eux-mêmes l'ordre de démarrage et d'arrêt grâce aux champs Required-Start et Required-Stop. Les services réseaux et remote_fs doivent être démarrés avant sshd. Le service démarre aux niveaux 3 et 5 et est stoppé aux niveaux 0 (arrêt), 1 (single user), 2 (sans réseau) et 6 (reboot).

BEGIN INIT INFO

Provides: sshd

Required-Start: $network $remote_fs

Required-Stop: $network $remote_fs

Default-Start: 3 5

Default-Stop: 0 1 2 6

Description: Start the sshd daemon

END INIT INFO

Voici la liste des options de chkconfig :

--list : liste de l'ensemble de la configuration.

--list service : la configuration d'un service donné.

--add service : ajoute le service indiqué dans la configuration System V.

--del service : supprime le service de la configuration System V.

--level xxx service **on/off** : active ou désactive le service pour les niveaux d'exécution indiqués.

```
# chkconfig --list

rwhod      0:arrêt  1:arrêt  2:arrêt  3:arrêt  4:arrêt  5:arrêt  6:arrêt

atd        0:arrêt  1:arrêt  2:arrêt  3:marche 4:marche 5:marche 6:arrêt

snmpd      0:arrêt  1:arrêt  2:marche 3:marche 4:marche 5:marche 6:arrêt

ntpd       0:arrêt  1:arrêt  2:marche 3:marche 4:marche 5:marche 6:arrêt

keytable   0:arrêt  1:marche 2:marche 3:marche 4:marche 5:marche 6:arrêt

syslog     0:arrêt  1:arrêt  2:marche 3:marche 4:marche 5:marche 6:arrêt

...


# chkconfig --list smb

smb        0:arrêt 1:arrêt 2:arrêt 3:arrêt 4:arrêt 5:arrêt 6:arrêt


# chkconfig --level 35 smb on

# chkconfig --list smb

smb        0:arrêt  1:arrêt  2:arrêt  3:marche 4:arrêt  5:marche 6:arrêt


# /sbin/chkconfig --add httpd
```



chkconfig ne lance aucun service. Il ne fait que paramétrer les niveaux d'exécution. Pour lancer un service, on utilisera le script associé ou la commande service.

Sous Debian

La commande **update-rc.d** crée les liens nécessaires dans les divers répertoires rc**n**.d, un peu comme chkconfig, mais de manière plus « brute » : les scripts n'ont pas d'information particulière dans leur en-tête et la commande ne fonctionne qu'au niveau du système de fichiers, mettant en place les divers liens selon vos indications.

Voici deux exemples de syntaxe. La première inscrit un service avec des paramètres par défaut. Dans ce cas le service est configuré pour démarrer sur les niveaux de 2 à 5 et s'arrêter aux niveaux 0, 1 et 6. La position d'arrêt/relance est à 20.

update-rc.d ssh defaults

Dans le second exemple, le service est inséré avec des options complètes. Au démarrage le service est en position 10 sur les niveaux 3, 4 et 5. À l'arrêt le service est à la position 5 sur les niveaux 0, 1 et 6. N'oubliez pas les points.

update-rc.d ssh start 10 3 4 5 . stop 05 0 1 6 .

Le paramètre remove supprime les liens des divers répertoires. Cependant le script **/etc/init.d/xxx** associé doit lui-même ne plus exister. Dans le cas contraire, utilisez le paramètre **-f** pour forcer la suppression des liens (le script lui-même reste en place).

update-rc.d -f ssh remove

8. Consoles virtuelles

Les consoles virtuelles permettent d'obtenir des terminaux virtuels sur une machine. Elles sont définies dans **/etc/inittab**. Elles sont disponibles via les périphériques **/dev/ttyn** où n est le numéro de console.

La couche graphique n'est généralement pas installée ni lancée sur les serveurs en entreprise. Vous accédez au serveur soit directement (ou par kvm), soit par le réseau (ssh par exemple) ou encore via des solutions intégrées (ports d'administration du serveur, comme ILO sur les machines HP).

Passez d'une console à l'autre avec la séquence de touches [Alt][F**n**] (ex : [Alt][F2]) depuis la console ou [Ctrl][Alt][F**n**] depuis X Window.

Utilisez les touches [Alt][Flèche à droite] et [Alt][Flèche à gauche] pour passer à la console suivante ou précédente.

/dev/ttyn représente la console virtuelle n.

/dev/tty0 représente la console courante.

Comme il y a 12 touches de fonction, il peut y avoir par défaut 12 terminaux virtuels.

Cependant 6 « seulement » sont activés par défaut.

X Window est lancé par défaut sur la première console disponible, généralement la 7.

Les consoles sont lancées par inittab et par les processus **getty** ou **mingetty**. Ce sont les seules entrées de inittab ou le label a son importance : il correspond au numéro de la console. Notez l'utilisation de respawn. Comme c'est la console qui établit la demande de login puis est substituée par le shell, lorsque le shell se termine, un processus mingetty est automatiquement relancé pour accepter une nouvelle connexion.

1:2345:respawn:/sbin/mingetty tty1

2:2345:respawn:/sbin/mingetty tty2

3:2345:respawn:/sbin/mingetty tty3

4:2345:respawn:/sbin/mingetty tty4

5:2345:respawn:/sbin/mingetty tty5

6:2345:respawn:/sbin/mingetty tty6

9. Les logins

Une fois les terminaux (getty) lancés par init, un prompt attend la saisie du nom (le login) de l'utilisateur. Avant ce prompt le contenu du fichier **/etc/issue** est affiché. Le nom validé, le terminal exécute la commande **login** qui va demander la saisie du mot de passe. Si le mot de passe est correct (vérification dans **/etc/passwd** et **/etc/shadow** ou utilisation des modules PAM), alors login affiche le contenu de **/etc/motd** et exécute un shell (toujours défini dans /etc/passwd).

Notez que getty et login n'effectuent pas de fork : les processus lancés ne sont pas des fils mais se substituent au processus courant (API exec). Il n'y a pas de relations pères-fils entre les processus (getty->login->shell) mais chacun se substitue au précédent, et le processus garde le même PID. C'est ainsi que init sait quand une connexion est terminée.

Une fois la session terminée (fin du shell), init relance un terminal pour une nouvelle connexion (commande respawn).

C'est getty qui va permettre un bon fonctionnement du terminal de l'utilisateur, en s'adaptant aux divers paramètres de celui-ci (VT100, VT220, XTERM, CONSOLE...). De plus getty peut parfaitement écouter un port série et supporter une connexion modem (et lancer ensuite une session ppp par exemple).

début

init : crée un terminal fils

terminal : attente du nom d'utilisateur

saisie de l'utilisateur, lancement de login

login : saisie du mot de passe

concordance ?

non → login : sortie

oui → login : lancement du shell

shell : exécution des commandes

shell : sortie

Séquence d’ouverture de session via login

10. Arrêt

Plusieurs méthodes permettent d’arrêter proprement une machine sous Linux. Tout d’abord pour mémoire les arrêts sont aussi gérés par init avec les niveaux 0 et 6. Les deux sont en pratique quasiment identiques sauf pour la dernière action.

Runlevel 0 : l’ordinateur est électriquement éteint.

Runlevel 6 : l’ordinateur reboote.

C’est ainsi que la commande suivante éteint l’ordinateur :

init 0

Et que celle-ci le reboote :

init 6

Cependant la commande la plus correcte, la plus propre et la plus sécuritaire pour arrêter le système est **shutdown**. Shutdown appelle init, mais accepte des paramètres supplémentaires. Sa syntaxe base est **shutdown <param> <délai> <message>**

Les paramètres sont :

Paramètre	Action
-k	N’effectue pas le shutdown mais envoie le message à tout le monde.
-r	C’est un reboot.
-h	(halt) c’est un arrêt.
-f	Empêche l’exécution de fsck au boot.
-F	Force l’exécution de fsck au boot.
-c	Annule le shutdown sans délai, mais un message est possible.

Le délai peut être spécifié de différentes manières :

hh:mm : une heure précise.

+m : dans m minutes.

now : un alias pour +0, c’est-à-dire tout de suite.

L’exemple suivant programme un reboot pour dans 10 minutes avec un message d’avertissement.

shutdown -r +10 "Reboot pour maintenance dans 10 minutes"

Broadcast message from root (pts/2) (Fri Apr 4 15:00:34 2008):

Reboot pour maintenance dans 10 minutes

The system is going DOWN for reboot in 10 minutes!

L’exemple suivant annule le reboot.

shutdown -c "Maintenance annulée"

Shutdown cancelled.

Broadcast message from root (pts/2) (Fri Apr 4 15:02:21 2008):

Maintenance annulée

Les commandes **reboot** et **halt** sont appelées en fin d’init 6 et 0, respectivement. Si elles sont appelées dans un autre niveau que 6 ou 0, elles sont l’équivalent d’un appel à shutdown :

TP : Démarrage de Linux, services, noyau et périphériques - GRUB et le processus de boot

THEME 107 Imprimer

107.2 Gérer les imprimantes et files d'impression (1), 107.3 Imprimer des fichiers (1) et 107.4 Installer et configurer des imprimantes (1)

L'impression

1. Principe

Il existe trois standards d'impression sous Unix, un sous System V, un autre sous BSD et un dernier fédérateur.

Quel que soit le standard, le principe est le même. À chaque imprimante déclarée (généralement dans /etc/printcap) correspond une file d'attente (**queue**). L'ensemble de ces files d'attente est géré par un service indépendant. Ces deux principes permettent une impression multi-utilisateur (les travaux d'impression sont en file d'attente, **job queues**), et en réseau (le service peut être utilisé depuis une autre machine distante).

En règle générale toutes les imprimantes savent directement imprimer du texte brut ASCII en 80 colonnes. Pour imprimer des documents formatés ou des images, on peut utiliser un pilote. On parle en fait de **filtre d'impression**. Le filtre peut être un script ou un binaire qui récupère le flux entrant (texte, image, document, postscript...), l'identifie et à l'aide de traitements associés le transforme en langage compréhensible par l'imprimante (Postscript, PCL, Canon, Epson, WPS...).



Si vous avez le choix et les moyens n'hésitez pas à prendre une imprimante compatible Postscript qui est un gage de parfaite compatibilité. Le site linuxprinting.org dispose d'une base complète de compatibilité des imprimantes sous Linux.

Linux accepte les commandes issues des Unix de type System V et BSD. Pendant longtemps le sous-système d'impression était basé sur les services BSD et le démon **lpd**. Depuis quelques années, toutes les distributions se basent sur CUPS, rétro-compatible (pour les commandes en tout cas) avec les anciens systèmes d'impression.

2. System V

Les commandes de gestion des files d'attente et des impressions sous System V sont les suivantes :

lp [-dImprimante] [-nChiffre] fic1 : imprime le contenu du fichier fic1. L'option **-d** permet de choisir l'imprimante, **-n** le nombre d'exemplaires.

lpstat [-d] [-s] [-t] [-p] : informations sur l'impression. L'option **-d** affiche le nom de l'imprimante par défaut, **-s** un état résumé des imprimantes, **-t** la totalité des informations (statut, travaux...), **-p [liste]** uniquement les informations sur les imprimantes incluses dans la liste.

cancel [ids] [printers] [-a] [-e] : supprime les tâches d'impression ids des imprimantes printers. L'option **-a** supprime tous les travaux de l'utilisateur, **-e** tous les travaux (seulement pour l'administrateur).

On peut trouver les commandes **enable** et **disable** qui prennent comme paramètre le nom de la file d'attente, permettant d'en activer ou désactiver l'accès.

Le démon (ou daemon) s'appelle généralement **lpd** (**line printer daemon**) ou **lpsched** (**line printer scheduler**).

lpadmin permet d'administrer les services d'impression comme les files d'attentes associées à une imprimante et la file d'attente par défaut. Ex : **lpadmin -p queue1 -v imprimante-m modèle**.

lpadmin -x file : suppression de la file d'attente.

lpadmin -d file : définir la file d'attente par défaut.

lpadmin -p file -u allow:liste : autorisation d'imprimer pour les utilisateurs précisés.

lpadmin -p file -u deny:liste : interdiction d'imprimer pour les utilisateurs précisés.

lpshut arrête le service d'impression. Au redémarrage du démon, les impressions en cours au moment de l'arrêt sont reprises.

accept et **reject** permettent de valider une file d'attente pour l'impression ou de la fermer.

lpmove permet de transférer des requêtes d'impression d'une file d'attente vers une autre.

3. BSD

lpr [-Pimprimante] [-#copies] fic1 : imprime le contenu du fichier fic1. L'option **-P** permet de spécifier l'imprimante, **-#** le nombre de copies.

lpq [-Pimprimante] : indique l'état et la liste des travaux pour l'imprimante éventuellement spécifiée par l'option **-P**.

lprm [-Pimprimante] [-] [ids] : permet de supprimer un travail de l'imprimante spécifiée par l'option **-P**, l'option **-** supprime tous les travaux de l'utilisateur, ids représente une liste de travaux à supprimer.

La commande **lpc** est une sorte de petit shell permettant de contrôler les imprimantes et les travaux.

Le service s'appelle généralement **lpd**.

4. CUPS

a. Présentation

CUPS (**Common Unix Printing System**) est un système d'impression Unix, orienté réseau :

Basé sur le protocole **IPP** (**Internet Printing Protocol**) basé lui-même sur le protocole **HTTP/1.1**.

Simple d'utilisation, notamment grâce à une configuration et une administration centralisée depuis une interface HTTP, des règles de conversion basées sur les types MIME, et des fichiers de description d'imprimante standards (**PPD**, **PostScript Printer Description**).

CUPS reprend les commandes System V et BSD déjà abordées pour plus de simplicité.

Les traces des impressions sont disponibles au format **CLF** (**Common Log Format**) de serveur Web et exploitables par les mêmes outils.

CUPS est capable d'interagir avec les serveurs d'impression LPD pour garder une compatibilité ascendante.

CUPS dispose de sa propre API permettant de créer des interfaces utilisateur pouvant s'intégrer dans des environnement graphiques ou des interfaces d'administration.

Les pools d'impression permettent la redirection automatique des tâches.

L'authentification est possible par utilisateur, hôte ou certificat numérique.

Le service d'impression se nomme **cupsd**.

```
$ ps -ef |grep cupsd

root      3128     1  0 Apr29 ?        00:00:01 /usr/sbin/cupsd
```

Il n'y a pas besoin d'outils graphiques pour administrer un serveur CUPS bien que pour des raisons de facilité la plupart des distributions, voire des environnements graphiques, en proposent. CUPS propose une interface d'administration WEB directement accessible depuis le port 631 du serveur. L'interface fonctionne avec n'importe quel navigateur HTTP.

http://localhost:631

Le fichier de configuration est **/etc/cups/cupsd.conf**. L'équivalent du fichier **/etc/printcap** est présent dans **/etc/cups/printers.conf**.

<DefaultPrinter lj2100>

Info Laserjet 2100

DeviceURI socket://192.168.1.10:9100

State Idle

StateTime 1203806079

Accepting Yes

Shared Yes

JobSheets none none

QuotaPeriod 0

PageLimit 0

KLimit 0

OpPolicy default

ErrorPolicy stop-printer

</Printer>

b. Ajout d'une imprimante

Vous avez deux solutions pour ajouter une imprimante :

Éditer les fichiers à la main.

Passer par l'interface Web ou un outil de votre distribution.

Dans le premier cas, vous devez modifier le fichier printers.conf pour ajouter une section pour votre imprimante puis copier dans /etc/cups/ppd le fichier PPD correspondant à votre imprimante et le renommer en utilisant le nom de section (ex : lj2100.ppd) du fichier printers.conf. Enfin vous devez rechercher la configuration de CUPS. Sous Red Hat ou openSUSE par exemple :

service cups reload

Comme l'interface Web est généralement activée par défaut, vous pouvez passer par un navigateur Web. Il est possible que lors de l'accès aux pages d'administration des identifiants

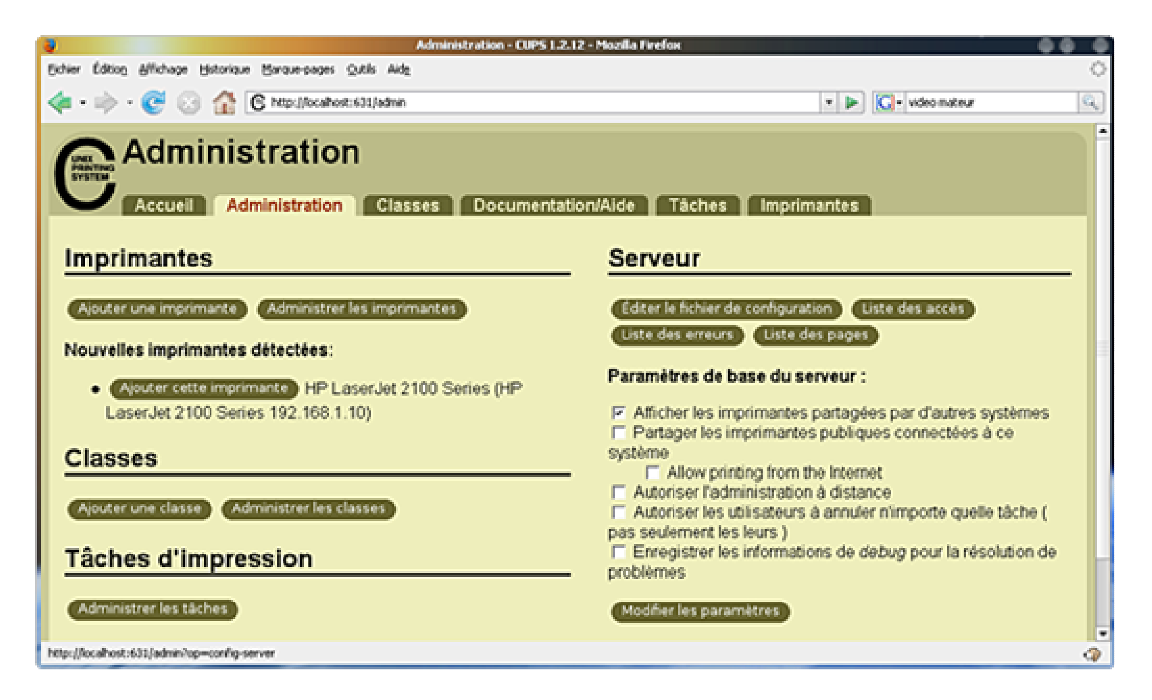
vous soient demandés. En principe ceux de root suffisent mais vous pouvez créer des comptes (ou rajouter des utilisateurs) chargés de la gestion des impressions avec la commande **lppasswd**.

lppasswd -a seb

Enter password:

Enter password again:

Sur la page principale cliquez sur l'onglet **Administration**.



Accueil du frontend Web de CUPS

Il est possible que CUPS détecte les imprimantes locales ou sur le réseau lorsqu'elles proposent des services LPD (port 9100) ou IPP car le service est à l'écoute en permanence. Dans l'exemple en cours une imprimante a été détectée et le fait de cliquer sur **Ajouter cette imprimante** simplifie grandement la tâche.

Pour l'exemple cependant, basé sur l'ajout manuel d'une imprimante HP Laserjet 2100, vous allez ajouter une imprimante en cliquant sur **Ajouter une imprimante**.

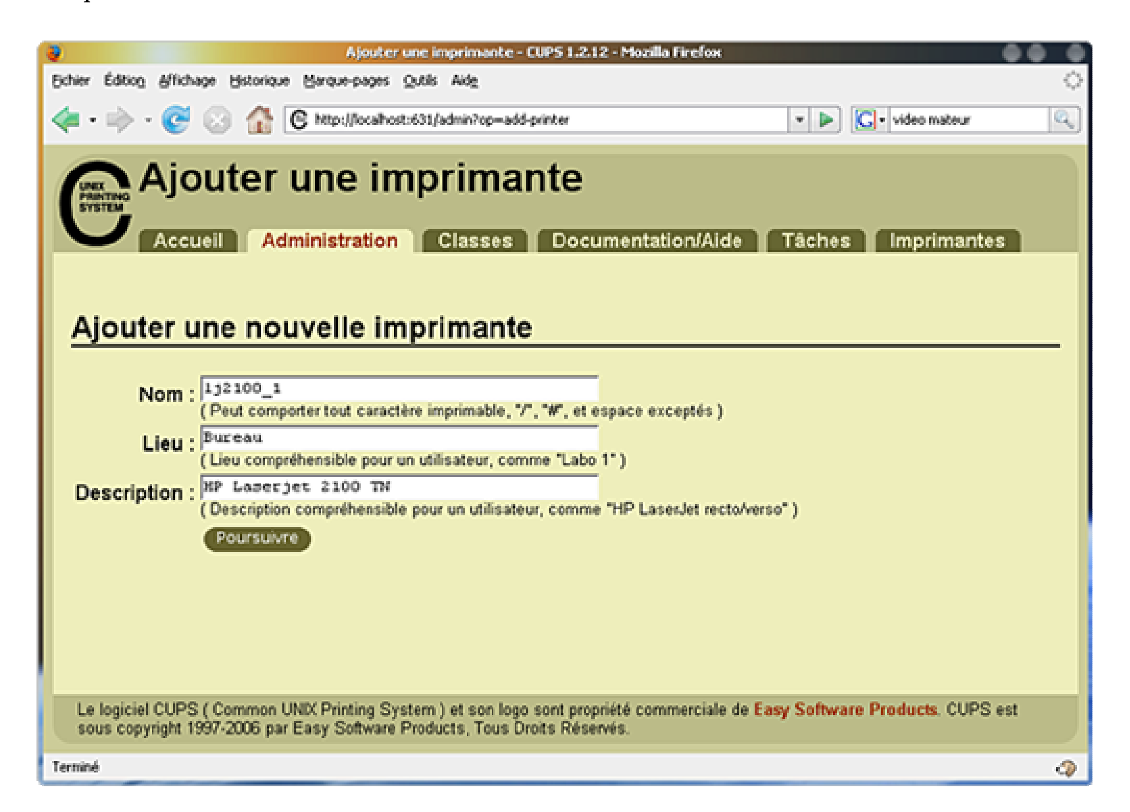
Saisissez les informations suivantes :

Le **nom** de l'imprimante est le nom de la file qui sera visible dans les outils d'impression. Vous ne devez pas mettre d'espaces.

Le **lieu** peut être laissé vide mais est utile pour la localiser (dans le cas d'une imprimante distante).

La **description** est ce que vous voulez, par exemple le modèle complet de l’imprimante.

Cliquez ensuite sur **Poursuivre**.



Ajouter un imprimante, première étape

L’étape suivante consiste à choisir votre type de connexion dans le menu matériel. L’imprimante est connectée en réseau. Dans ce cas sélectionnez **AppSocket/HP JetDirect**. Cliquez sur **Poursuivre**.



Ajouter une imprimante sur un port réseau

Pour une connexion réseau, vous avez plusieurs possibilités. La plupart des imprimantes connectées à un réseau de type Ethernet ou Wi-Fi proposent des services d'impression LPD ou Socket (impression directe). Dans ce dernier cas c'est l'imprimante elle-même qui gère les tâches d'impression. Pour les autres (IPP, http, Samba, etc.), veuillez consulter le mode d'emploi de votre imprimante. Quelques imprimantes professionnelles proposent une interface de configuration Web (comme les routeurs grands public) pour paramétrer les ports.

Si votre imprimante distante est déjà configurée sur un autre serveur CUPS, vous pouvez passer par ce serveur et les **URI** de type http, ipp ou lpd pour imprimer dessus.

L'imprimante de test Laserjet 2100 dispose d'une carte réseau et d'un serveur d'impression jetDirect intégré. L'URI saisie est **socket://192.168.1.10:9100**.

L'IP est celle de l'imprimante sur le réseau, le port 9100 étant le port standard dans ce cas.

Cliquez ensuite sur **Poursuivre**.



Adresse et port réseau de l'imprimante

Choisissez maintenant un pilote correspondant à votre modèle d'imprimante. Vous remarquerez que pour plusieurs modèles il existe plusieurs pilotes. Celui recommandé est généralement indiqué comme tel (**recommanded**). Vous devriez vérifier quel pilote est réellement recommandé pour votre imprimante sur le site suivant :

http://www.linux-foundation.org/en/OpenPrinting

Qui informe que pour cette imprimante :

For basic printing functionality use the Postscript PPD. For advanced functionality such as printer status and maintenance features, use the **HPLIP** driver (which includes **HPIJS**).

Le choix se porte donc sur le pilote **HP Laserjet 2100 Foomatic/hpijs**.



Les imprimantes (et les scanners) de marque HP sont particulièrement bien supportés sous Linux au travers du projet libre de l'éditeur **HPLIP** (**HP Linux Imaging and Printing**), y compris les solutions intégrées (imprimante, scanner, fax, photocopieur, impression photo, etc.). La liste du matériel supporté est accessible via http://hplip.sourceforge.net/.

Vous pouvez préciser un autre fichier PPD. Plusieurs constructeurs proposent ce genre de fichiers pour leur imprimante. Les fichiers PPD sont présents par défaut dans /usr/share/cups/model.

Cliquez sur **Ajouter une imprimante**.

Ajouter une imprimante - CUPS 1.2.12 - Mozilla Firefox

Fichier Édition Affichage Historique Marque-pages Outils Aide

http://localhost:631/admin

Ajouter une imprimante

Accueil Administration Classes Documentation/Aide Tâches Imprimantes

Modèle/Pilote pour lj2100_1

Modèle:
HP LaserJet 2100 - CUPS+Gutenprint v5.0.1 Simplified (en)
HP LaserJet 2100 - CUPS+Gutenprint v5.0.1 Simplified (en)
HP LaserJet 2100 Foomatic/hpijs (en)
HP LaserJet 2100 Foomatic/lj4dith (en)
HP LaserJet 2100 Foomatic/lj5gray (en)
HP LaserJet 2100 Foomatic/ljet4 (en)
HP LaserJet 2100 Foomatic/pxlmono (en)
HP LaserJet 2100 Series Postscript (en)
HP LaserJet 2100 Series Postscript (en)
HP LaserJet 2100M - CUPS+Gutenprint v5.0.1 (en)

Ou donnez un fichier PPD : Parcourir...

Ajouter une imprimante

Le logiciel CUPS (Common UNIX Printing System) et son logo sont propriété commerciale de Easy Software Products. CUPS est sous copyright 1997-2006 par Easy Software Products, Tous Droits Réservés.

Terminé

Sélection du pilote d'impression

Vous arrivez sur l'écran présentant la liste des imprimantes actuellement configurées. Vous pouvez tester votre imprimante en imprimant une page de test. Vous pouvez aussi modifier les options par défaut de l'imprimante. Ces options sont de plusieurs types :

Options générales pour indiquer le type de papier par défaut (A4), la qualité de sortie, le bac par défaut (les Laserjet 2100TN ont deux bacs et une entrée en façade), l'impression en Duplex (si le kit est présent), etc.

Le mode de sortie ou les options diverses, pour régler par exemple la résolution et le type de papier par défaut, un filtrage quelconque, etc.

Les bannières : pour séparer les impressions (cas des liasses par exemple) vous pouvez placer des bannières avant et après l'impression. Les fichiers des bannières peuvent être personnalisés dans certaines limites (voyez le manuel de CUPS pour cela).

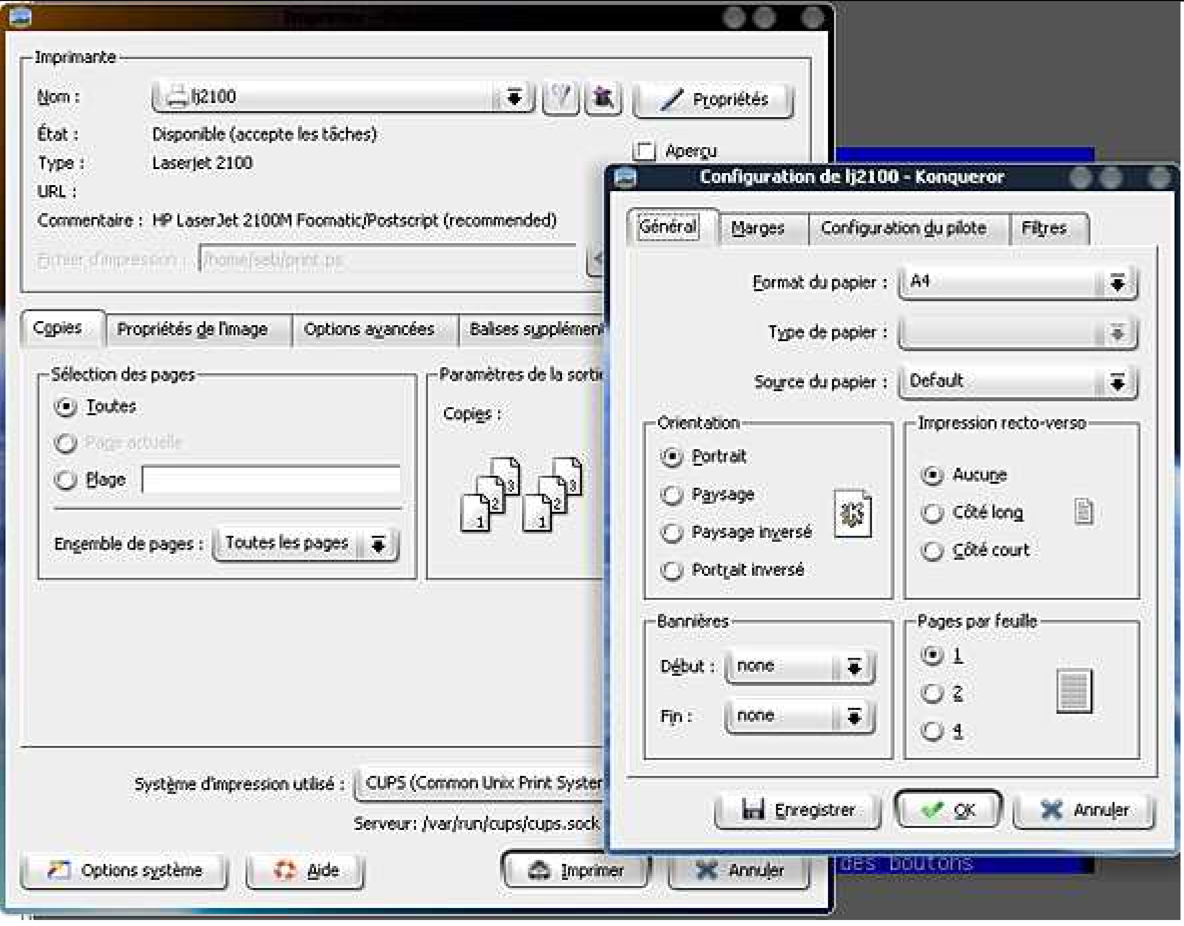
La facturation. CUPS peut être configuré pour gérer la facturation par service/machines/utilisateurs, dans le cas de l'utilisation de certains services avancés.

Réglages de l'imprimante

Vous devriez laisser, outre le format de papier, les options par défaut. La plupart des environnements de bureau et des logiciels proposent de modifier les options d'impression via CUPS comme sur les systèmes de type Windows.

La capture suivante a été effectuée sous KDE. Dans **Konqueror** l'entrée **Imprimer** du menu **Fichier** a été appelée lors de l'affichage d'une image. Toutes les options accessibles par le bouton **Propriétés** à côté du nom de l'imprimante sont issues des possibilités offertes par CUPS et le pilote d'impression PPD. D'une imprimante à l'autre et même d'un pilote à l'autre (si plusieurs sont disponibles) vous ne bénéficierez pas des mêmes options.

Si vous cliquez sur l'onglet **Configuration du pilote** vous obtiendrez le même choix d'options que ce que l'interface Web de CUPS vous propose.



Boîtes de dialogue d'impression sous KDE

Les distributions et les environnements graphiques proposent souvent leurs propres outils d'impression ou de gestion des imprimantes. Ainsi avec **KDE**, rendez-vous dans le **Centre de configuration**, **Périphériques** puis **Imprimantes**. Vous avez dès lors accès à toutes les imprimantes connues sur le système et même plus puisque certaines pseudo-imprimantes (qui ne sont pas gérées par CUPS) permettent d'imprimer dans des fichiers en PDF, d'envoyer une impression par mail, d'envoyer un fax, etc.

En cliquant sur **Mode superutilisateur** en bas à gauche et en saisissant votre mot de passe root, vous pouvez ajouter des imprimantes via le bouton associé en haut à gauche. Les étapes reprennent de manière mieux intégrée celles de CUPS.

TP : Les tâches administratives - L'impression

THEME 108 Documentation

108.3 Informer les utilisateurs (1)

Administration des utilisateurs

Notifications à l’utilisateur

a. /etc/issue

Lorsqu’un utilisateur se connecte depuis la console, un message est généralement affiché juste avant l’invite de saisie de son login. Ce message est contenu dans le fichier **/etc/issue**. C’est un message d’accueil et à ce titre il peut contenir tout ce que vous voulez. Par défaut, il contient généralement le nom de la distribution Linux et le numéro de version du noyau.

Voici l’exemple du contenu de **/etc/issue** sur une distribution Mandriva :

$ cat issue

Mandriva Linux release 2008.1 (Official) for i586

Kernel 2.6.24.4-desktop-1mnb on an i686 / \l

b. /etc/issue.net

Le message d’accueil peut être différent lorsqu’un utilisateur se connecte depuis une console distante (telnet, ssh, etc.). C’est souvent le même mais sans les caractères de contrôles liés à un shell donné. Pour modifier ce message spécifique, éditez le contenu du fichier **/etc/issue.net**.

c. /etc/motd

Motd signifie Message of the day, le message du jour. Une fois l’utilisateur connecté depuis une console (locale ou distante), un message peut être affiché. L’administrateur peut modifier ce message en éditant le fichier **/etc/motd**. Par défaut il est vide. Vous pouvez par exemple modifier ce fichier pour prévenir vos utilisateurs qu’un reboot de maintenance aura lieu tel jour à telle heure, ceci évitant d’envoyer n mails…

TP : Les tâches administratives - Gestion des utilisateurs

THEME 109 Shell, scripts, programmation et compilation

109.1 Modifier l’environnement du shell (5)

L’environnement utilisateur

a. /etc/skel

À la création d’un utilisateur et de son répertoire personnel, l’environnement de l’utilisateur est mis en place. L’environnement contient par exemple les variables d’environnement, les alias, l’exécution de divers scripts. Il est contenu dans des fichiers chargés au démarrage de l’interpréteur de commandes (shell).

Lors de la création d’un compte, les divers fichiers de configuration sont copiés depuis le contenu du répertoire /etc/skel (skeleton) vers le répertoire personnel. Si vous souhaitez modifier les environnements de façon globale AVANT la création des utilisateurs, vous pouvez placer dans /etc/skel tous les fichiers que vous souhaitez et les modifier selon votre convenance. Ainsi si par exemple vous souhaitez que tout le monde dispose des mêmes icônes par défaut sur son bureau, et la même configuration par défaut du bureau, placez-y les répertoires **Desktop** et **.kde** d’un compte modèle.

b. Scripts de configuration

Le chapitre Le shell et les commandes GNU vous a présenté les différents fichiers dont le contenu est exécuté par le shell. À la connexion d’un utilisateur, les scripts suivants sont exécutés dans cet ordre :

/etc/profile : définit les variables d’environnement importantes comme PATH, LOGNAME, USER, HOSTNAME, HISTSIZE, MAIL et INPUTRC.

/etc/profile.d/* : /etc/profile appelle tous les scripts présents dans ce répertoire. Ces scripts peuvent compléter la configuration globale en ajoutant par exemple la configuration des paramètres linguistiques, des alias globaux, etc.

~/.bash_profile : si le shell est bash, c’est le script suivant à être exécuté. Il est dans le répertoire utilisateur et appelle un autre script : **~/.bashrc** qui appelle lui-même **/etc/bashrc**. Vous pouvez définir dans .bash_profile des variables supplémentaires, alors que vous aurez tendance à définir dans **~/.bashrc** des alias et des fonctions. Il n’y a pas de règles strictes.

/etc/bashrc est utilisé pour définir les fonctions et alias pour tout le système et tous les utilisateurs sous bash.

c. Groupes privés et setgid

La politique de Red Hat et d'autres distributions associées, comme Fedora, pour la sécurité des utilisateurs est de systématiquement leur créer un groupe privé et de leur attribuer un masque 002. De ce fait, les fichiers sont mieux protégés par défaut puisqu'aucun groupe ne peut accéder aux fichiers par défaut : les fichiers créés n'appartiennent qu'à un groupe qui n'a qu'un membre.

Comme un utilisateur peut faire partie de plusieurs groupes, il peut en théorie étendre l'accès à ses fichiers et répertoires avec la commande **chgrp** qui change le groupe d'un fichier. De même il peut changer temporairement de groupe principal avec la commande **newgrp**.

Cependant l'utilisateur préfère souvent ne pas se « casser la tête » et donner tous les droits avec un **chmod** (ex. chmod 777) ce qui constitue une faille dans la sécurité.

Dans ce cas, la solution est d'utiliser le bit **setgid** sur un répertoire.

On définit un groupe commun à tous les utilisateurs devant pouvoir accéder à un répertoire et à ses fichiers.

On crée un répertoire dont le groupe propriétaire est le groupe commun.

On donne au répertoire les droits avec l'umask 002 (**rwxrwxr-x**) ou autre, mais avec tous les droits sur le groupe.

On ajoute au répertoire le droit setgid (s) sur le groupe.

mkdir rep

chgrp grpcommun rep

chmod 2770 rep

Lorsque setgid est positionné sur un répertoire, tous les fichiers créés dans ce répertoire appartiennent au groupe du répertoire et non pas au groupe de l'utilisateur. Ainsi tous les membres du groupe commun pourront accéder aux données. Le fichier continue cependant à appartenir à son créateur.

Associé au Sticky Bit, vous obtenez un bon niveau de protection et d'accès aux données :

tous les membres du groupe peuvent y créer des fichiers et des répertoires,

tous les fichiers appartiennent automatiquement au même groupe,

seuls les propriétaires des fichiers peuvent les supprimer.

chmod 3770 rep

ls -ld rep

drwxrws--T 2 seb grpcommun 1024 2008-04-24 11:39 rep/

$ ls -l test

-rw-r--r-- 1 seb grpcommun 0 2008-04-24 11:43 test

TP : Les tâches administratives - Gestion des utilisateurs

Configuration de bash

1. Fichiers de configuration

Le shell bash peut être lancé dans plusieurs modes :

le shell interactif de connexion (login shell) ;

le shell interactif simple ;

le shell non interactif ;

le mode sh ;

etc.

Selon son mode de démarrage, le shell va chercher et exécuter divers scripts et fichiers de configuration. Un fichier de configuration est un script shell, une séquence de commandes individuelles ayant pour but de configurer l'environnement de l'utilisateur.

a. Shell de connexion

Le shell de connexion est lancé après la saisie du login et du mot de passe sur la console. C'est celui précisé à la fin de chaque ligne de /etc/passwd. Dans ce mode, le shell cherche à exécuter, dans cet ordre et s'ils sont présents :

/etc/profile

~/.bash_profile

~/.bash_login

~/.profile

À la déconnexion, il tente d'exécuter :

- ~/.bash_logout

b. Shell simple

Le shell interactif simple correspond à l'exécution du bash dans une fenêtre (xterm, konsole), une console ou à la main (taper bash dans une console). Dans ce cas seul le fichier suivant sera exécuté s'il existe :

~/.bashrc

Notez que dans beaucoup de distributions Linux, le .bashrc est appelé soit par .bash_profile, soit par /etc/profile, et que la configuration est donc placée dans .bash_profile qui sera alors toujours appelé.

c. Mode Bourne shell

Lorsque bash est lancé en mode Bourne Shell via la commande **sh**, en shell de connexion ou non, il tente d'exécuter les fichiers dans cet ordre :

/etc/profile

~/.profile

d. Mode non interactif

Le shell peut être lancé en mode non interactif. C'est généralement le cas lorsque vous exécutez un script. Dans ce cas il n'y a aucun script exécuté par défaut au démarrage sauf si vous précisez une variable appelée BASH_ENV qui contient le chemin d'un script. Dans ce cas bash charge et exécute ce fichier avant le début de l'exécution du script ou de la commande.

2. Commandes set

Le shell dispose de dizaines d'options, dont la plupart peuvent être paramétrées à l'aide de la commande **set**. Celles qui suivent ne sont qu'une simple sélection. Le - avant l'option permet de passer celle-ci à On. Un + passe l'option à off.

Option	Résultat
-a / -o allexport	Toutes les variables seront automatiquement exportées.
-u / -o nounset	Par défaut le shell traite les variables inexistantes comme des chaînes vides. Cette option produit une erreur si la variable n'existe pas.
-x / -o xtrace	Affiche toutes les commandes au fur et à mesure de leur exécution : idéal en début de script pour débuggage.
-o vi	Manipulation de la ligne de commande avec la syntaxe de vi.
-o emacs	Manipulation de la ligne de commande avec la syntaxe de emacs.
-C / -o noclobber	Interdit les redirections en sortie si le fichier existe déjà.
history	Autorise la gestion de l'historique.

Le manuel du shell vous fournira toutes les options possibles.

109.2 Ecrire et modifier des scripts (3)

1. Structure et exécution d'un script

Le shell n'est pas qu'un simple interpréteur de commandes, mais dispose d'un véritable langage de programmation avec notamment une gestion des variables, des tests et des boucles, des opérations sur les variables, des fonctions...

Toutes les instructions et commandes sont regroupées au sein d'un script. Lors de son exécution, chaque ligne sera lue une à une et exécutée. Une ligne peut se composer de commandes internes ou externes, de commentaires ou être vide. Plusieurs instructions par lignes sont possibles, séparées par le **;** ou liées conditionnellement par **&&** ou **||**. Le **;** est l'équivalent d'un **saut de ligne**.

Par convention les noms des shell scripts se terminent généralement (pas obligatoirement) par « .sh » pour le Bourne Shell et le Bourne Again Shell, par « .ksh » pour le Korn Shell et par « .csh » pour le C Shell.

Pour rendre un script exécutable directement :

$ chmod u+x monscript

Pour l'exécuter :

$ ./monscript

Pour éviter le ./ :

$ PATH=$PATH:.

$ monscript

Notez que le point est placé en dernier dans le PATH. Le mettre en premier peut présenter un risque pour la sécurité : une nouvelle commande **ls** modifiée est placée dans votre répertoire.

Quand un script est lancé, un nouveau shell fils est créé qui va exécuter chacune des commandes. Si c'est une commande interne, elle est directement exécutée par le nouveau shell. Si c'est une commande externe, dans le cas d'un binaire un nouveau fils sera créé pour l'exécuter, dans le cas d'un shell script un nouveau shell fils est lancé pour lire ce nouveau shell ligne à ligne.

Une ligne de commentaire commence toujours par le caractère **#**. Un commentaire peut être placé en fin d'une ligne comportant déjà des commandes.

La ligne suivante effectue un ls

ls # La ligne en question

La première ligne a une importance particulière car elle permet de préciser quel shell va exécuter le script :

#!/bin/bash

#!/bin/ksh

Dans le premier cas c'est un script Bourne Again, dans l'autre un script Korn.

2. Arguments d'un script

a. Paramètres de position

Les paramètres de position sont aussi des variables spéciales utilisées lors d'un passage de paramètres à un script.

Variable	Contenu
$0	Nom de la commande (du script).
$1-9	$1,$2,$3... Les neuf premiers paramètres passés au script.
$#	Nombre total de paramètres passés au script.
$*	Liste de tous les paramètres au format "$1 $2 $3 ...".
$@	Liste des paramètres sous forme d'éléments distincts "$1" "$2" "$3" ...

```
$ cat param.sh

#!/bin/bash



echo "Nom : $0"

echo "Nombre de parametres : $#"

echo "Parametres : 1=$1 2=$2 3=$3"

echo "Liste : $*"

echo "Elements : $@"



$ param.sh riri fifi loulou

Nom : ./param.sh

Nombre de parametres : 3

Parametres : 1=riri 2=fifi 3=loulou
```

Liste : riri fifi loulou

Elements : riri fifi loulou

La différence entre **$@** et **$*** ne saute pas aux yeux. Reprenez l'exemple précédent avec une petite modification :

$ param.sh riri "fifi loulou"

Nom : ./param.sh

Nombre de parametres : 2

Parametres : 1=riri 2=fifi loulou 3=

Liste : riri fifi loulou

Elements : riri fifi loulou

Cette fois-ci il n'y a que deux paramètres de passés. Pourtant les listes semblent visuellement identiques. En fait si la première contient bien :

"riri fifi loulou"

La deuxième contient :

"riri" "fifi loulou"

Soit bien deux éléments. Dans le premier exemple vous aviez :

"riri" "fifi" "loulou"

b. Redéfinition des paramètres

Outre le fait de lister les variables, l'instruction **set** permet de redéfinir le contenu des variables de position. Avec :

set valeur1 valeur2 valeur3 ...

$1 prendra comme contenu valeur1, $2 valeur2 et ainsi de suite.

$ cat param2.sh

#!/bin/bash

echo "Avant :"

echo "Nombre de parametres : $#"

echo "Parametres : 1=$1 2=$2 3=$3 4=$4"

echo "Liste : $*"

set alfred oscar romeo zoulou

echo "apres set alfred oscar romeo zoulou"

echo "Nombre de parametres : $#"

echo "Parametres : 1=$1 2=$2 3=$3 4=$4"

echo "Liste : $*"

$ ./param2.sh riri fifi loulou donald picsou

Avant :

Nombre de parametres : 5

Parametres : 1=riri 2=fifi 3=loulou 4=donald

Liste : riri fifi loulou donald picsou

apres set alfred oscar romeo zoulou

Nombre de parametres : 4

Parametres : 1=alfred 2=oscar 3=romeo 4=zoulou

Liste : alfred oscar romeo zoulou

c. Réorganisation des paramètres

La commande **shift** est la dernière commande permettant de modifier la structure des paramètres de position. Un simple appel décale tous les paramètres d’une position en supprimant le premier : $2 devient $1, $3 devient $2 et ainsi de suite. Le $1 originel disparaît. $#, $* et $@ sont redéfinis en conséquence.

La commande **shift** suivie d’une valeur n effectue un décalage de n éléments. Ainsi avec **shift 4** $5 devient $1, $6 devient $2, ...

$ cat param3.sh

#!/bin/bash

set alfred oscar romeo zoulou

echo "set alfred oscar romeo zoulou"

echo "Nombre de parametres : $#"

echo "Parametres : 1=$1 2=$2 3=$3 4=$4"

echo "Liste : $*"

shift

echo "Après un shift"

echo "Nombre de parametres : $#"

echo "Parametres : 1=$1 2=$2 3=$3 4=$4"

echo "Liste : $*"

$ ./param3.sh

set alfred oscar romeo zoulou

Nombre de parametres : 4

Parametres : 1=alfred 2=oscar 3=romeo 4=zoulou

Liste : alfred oscar romeo zoulou

Après un shift

Nombre de parametres : 3

Parametres : 1=oscar 2=romeo 3=zoulou 4=

Liste : oscar romeo zoulou

d. Sortie de script

La commande **exit** permet de mettre fin à un script. Par défaut la valeur retournée est 0 (pas d'erreur) mais n'importe quelle autre valeur de 0 à 255 peut être précisée. Vous récupérez la valeur de sortie par la variable $?.

$ exit 1

3. Environnement du processus

En principe seules les variables exportées sont accessibles par un processus fils. Si vous souhaitez visualiser l'environnement lié à un fils (dans un script par exemple) utilisez la commande **env**.

```
$ env

LESSKEY=/etc/lesskey.bin

NNTPSERVER=news

INFODIR=/usr/local/info:/usr/share/info:/usr/info

MANPATH=/usr/local/man:/usr/share/man

KDE_MULTIHEAD=false

SSH_AGENT_PID=28012

HOSTNAME=slyserver

DM_CONTROL=/var/run/xdmctl

XKEYSYMDB=/usr/share/X11/XKeysymDB

HOST=p64p17bicb3

SHELL=/bin/bash

TERM=xterm

PROFILEREAD=true

HISTSIZE=1000

XDM_MANAGED=/var/run/xdmctl/xdmctl-

:0,maysd,mayfn,sched,rsvd,method=classic

XDG_SESSION_COOKIE=16af07a56781b46897182100470603OO-

1211264847.394692-546885666

TMPDIR=/tmp

GTK2_RC_FILES=/etc/gtk-2.0/gtkrc:/usr/share/themes//QtCurve/gtk-

2.0/gtkrc:/home/seb/.gtkrc-2.0-qtengine:/home/seb/.gtkrc-

2.0:/home/seb/.kde/share/config/gtkrc-2.0
```

KDE_NO_IPV6=1

GTK_RC_FILES=/etc/gtk/gtkrc:/home/seb/.gtkrc:/home/seb/.kde/share/co

nfig/gtkrc

GS_LIB=/home/seb/.fonts

WINDOWID=71303176

MORE=-sl

QTDIR=/usr/lib/qt3

XSESSION_IS_UP=yes

KDE_FULL_SESSION=true

GROFF_NO_SGR=yes

JRE_HOME=/usr/lib/jvm/jre

USER=seb

...

La commande **env** permet de redéfinir aussi l'environnement du processus à lancer. Cela peut être utile lorsque le script doit accéder à une variable qui n'est pas présente dans l'environnement du père, ou qu'on ne souhaite pas exporter. La syntaxe est :

env var1=valeur var2=valeur ... commande

Dans le cas de bash, env n'est pas indispensable.

var1=valeur var2=valeur ... commande

Si la première option est le signe - alors c'est tout l'environnement existant qui est supprimé pour être remplacé par les nouvelles variables et valeurs.

$ unset a

$ ./voir_a.sh

a=

$ env a=jojo ./voir_a.sh

a=jojo

$ echo a=$a

a=

4. Substitution de commande

Le mécanisme de substitution permet de placer le résultat de commandes simples ou complexes dans une variable. Vous placez les commandes à exécuter entre des accents graves « ` » ([Alt Gr] **7**).

$ mon_unix=`uname`

$ echo ${mon_unix}

Linux

$ machine=`uname -a | cut -d" " -f5`

echo $machine

SMP

Attention, seul le canal de sortie standard est affecté à la variable. Le canal d'erreur standard sort toujours vers l'écran dans ce cas.

Les accents graves ne sont pas toujours idéaux pour ces manipulations. En effet si vous travaillez à plusieurs niveaux, vous devez verrouiller ceux qui sont à l'intérieur des premiers niveaux. Aussi le bash permet d'utiliser à la place la syntaxe **$(...)** qui n'a pas ce problème.

$ mon_unix=$(uname)

$ echo ${mon_unix}

Linux

$ machine=$(uname -a | cut -d" " -f5)

echo $machine

SMP



Ne confondez pas les accolades et les parenthèses. Les premières isolent les variables, les secondes effectuent la substitution des commandes.

5. Tests de conditions

La commande **test** permet d'effectuer des tests de conditions. Le résultat est récupérable par la variable **$?** (code retour). Si ce résultat est 0 alors la condition est réalisée.

a. Tests sur une chaîne

test -z "variable" : zero, retour OK si la variable est vide (ex : test -z "$a").

test -n "variable" : non zero, retour OK si la variable n'est pas vide (texte quelconque).

test "variable" = chaîne : OK si les deux chaînes sont identiques.

test "variable" ! = chaîne : OK si les deux chaînes sont différentes.

$ a=

$ test -z "$a" ; echo $?

0

$ test -n "$a" ; echo $?

1

$ a=Jules

$ test "$a" = Jules ; echo $?

0

Attention à bien placer vos variables contenant du texte entre guillemets. Dans le cas contraire un bug se produira si la variable est vide :

$ a=

$ b=toto

$ [$a = $b] && echo "ok"

bash: [: =: unary operator expected

Alors que :

["$a" = "$b"] && echo "ok"

ne produit pas d'erreur.

b. Tests sur les valeurs numériques

Les chaînes à comparer sont converties en valeurs numériques. Bash ne gère que des valeurs entières. La syntaxe est :

test valeur1 option valeur2

et les options sont les suivantes :

Option	Rôle
-eq	Equal : égal
-ne	Not Equal : différent
-lt	Less than : inférieur
-gt	Greater than : supérieur
-le	Less or equal : inférieur ou égal
-ge	Greater or equal : supérieur ou égal

$ a=10

$ b=20

$ test "$a" -ne "$b" ; echo $?

0

$ test "$a" -ge "$b" ; echo $?

1

$ test "$a" -lt "$b" && echo "$a est inferieur a $b"

10 est inferieur a 20

c. Tests sur les fichiers

La syntaxe est :

test option nom_fichier

et les options sont les suivantes :

Option	Rôle
-f	Fichier normal.
-d	Un répertoire.
-c	Fichier en mode caractère.
-b	Fichier en mode bloc.
-p	Tube nommé (named pipe).

Option Rôle

-r Autorisation en lecture.

-w Autorisation en écriture.

-x Autorisation en exécution.

-s Fichier non vide (au moins un caractère).

-e Le fichier existe.

-L Le fichier est un lien symbolique .

-u Le fichier existe, SUID-Bit positionné.

-g Le fichier existe SGID-Bit positionné.

```
$ ls -l

-rw-r--r--   1 seb   users     1392 Aug 14 15:55 dump.log

lrwxrwxrwx   1 seb   users        4 Aug 14 15:21 lien_fic1 -> fic1

lrwxrwxrwx   1 seb   users        4 Aug 14 15:21 lien_fic2 -> fic2

-rw-r--r--   1 seb   users      234 Aug 16 12:20 liste1

-rw-r--r--   1 seb   users      234 Aug 13 10:06 liste2

-rwxr--r--   1 seb   users      288 Aug 19 09:05 param.sh

-rwxr--r--   1 seb   users      430 Aug 19 09:09 param2.sh

-rwxr--r--   1 seb   users      292 Aug 19 10:57 param3.sh

drwxr-xr-x   2 seb   users     8192 Aug 19 12:09 rep1

-rw-r--r--   1 seb   users     1496 Aug 14 16:12 resultat.txt

-rw-r--r--   1 seb   users     1715 Aug 16 14:55 toto.txt

-rwxr--r--   1 seb   users       12 Aug 16 12:07 voir_a.sh

$ test -f lien_fic1 ; echo $?

1

$ test -x dump.log ; echo $?

1

$ test -d rep1 ; echo $?
```

0

d. Tests combinés par des critères ET, OU, NON

Vous pouvez effectuer plusieurs tests avec une seule instruction. Les options de combinaison sont les mêmes que pour la commande **find**.

Critère	Action
-a	AND, ET logique
-o	OR, OU logique
!	NOT, NON logique
(...)	groupement des combinaisons. Les parenthèses doivent être verrouillées \(...\).

$ test -d "rep1" -a -w "rep1" && echo "rep1: repertoire, droit en

ecriture"

rep1: repertoire, droit en ecriture

e. Syntaxe allégée

Le mot test peut être remplacé par les crochets ouverts et fermés **[...]**. Il faut respecter un espace avant et après les crochets.

$ ["$a" -lt "$b"] && echo "$a est inferieur a $b"

10 est inferieur a 20

Le bash (et le ksh) intègre une commande interne de test qui se substitue au binaire test. Dans la pratique, la commande interne est entièrement compatible avec la commande externe mais bien plus rapide car il n'y a pas de création de nouveau processus. Pour forcer l'utilisation de la commande interne, utilisez les doubles crochets **[[...]]**.

$ [["$a" -lt "$b"]] && echo "$a est inferieur a $b"

10 est inferieur a 20

6. if ... then ... else

La structure **if then else fi** est une structure de contrôle conditionnelle.

if <commandes_condition>

then

<commandes exécutées si condition réalisée>

else

<commandes exécutées si dernière condition pas réalisée>

fi

Vous pouvez aussi préciser **elif**, en fait un else if. Si la dernière condition n'est pas réalisée, on en teste une nouvelle.

```
$ cat param4.sh

#!/bin/bash

if [ $# -ne 0 ]

then

        echo "$# parametres en ligne de commande"

else

        echo "Aucun parametre; set alfred oscar romeo zoulou"

        set alfred oscar romeo zoulou

fi



echo "Nombre de parametres : $#"

echo "Parametres : 1=$1 2=$2 3=$3 4=$4"

echo "Liste : $*"



$ ./param4.sh titi toto

2 parametres en ligne de commande

Nombre de parametres : 2

Parametres : 1=toto 2=titi 3= 4=

Liste : toto titi
```

$ ./param4.sh

Aucun parametre; set alfred oscar romeo zoulou

Nombre de parametres : 4

Parametres : 1=alfred 2=oscar 3=romeo 4=zoulou

Liste : alfred oscar romeo zoulou

7. Choix multiples case

La commande **case ... esac** permet de vérifier le contenu d’une variable ou d’un résultat de manière multiple.

case Valeur in

   Modele1) Commandes ;;

   Modele2) Commandes ;;

   *) action_defaut ;;

esac

Le modèle est soit un simple texte, soit composé de caractères spéciaux. Chaque bloc de commandes lié au modèle doit se terminer par deux points-virgules. Dès que le modèle est vérifié, le bloc de commandes correspondant est exécuté. L’étoile en dernière position (chaîne variable) est l’action par défaut si aucun critère n’est vérifié. Elle est facultative.

Caractère	Rôle
*	Chaîne variable (même vide)
?	Un seul caractère
[...]	Une plage de caractères
[!...]	Négation de la plage de caractères
|	OU logique

$ cat case1.sh

#!/bin/bash

if [$# -ne 0]

then

      echo "$# parametres en ligne de commande"

else

```
    echo "Aucun parametre; set alfred oscar romeo zoulou"
    exit 1
fi

case $1 in
    a*)
        echo "Commence par a"
        ;;
    b*)
        echo "Commence par b"
        ;;
    fic[123])
        echo "fic1 fic2 ou fic3"
        ;;
    *)
        echo "Commence par n'importe"
        ;;
esac

exit 0
$ ./case1.sh "au revoir"
Commence par a
$ ./case1.sh bonjour
Commence par b
$ ./case1.sh fic2
```

fic1 ou fic2 ou fic3

$ ./case1.sh erreur

Commence par n’importe

8. Saisie de l’utilisateur

La commande **read** permet à l’utilisateur de saisir une chaîne et de la placer dans une ou plusieurs variable. La saisie est validée par [Entrée].

read var1 [var2 ...]

Si plusieurs variables sont précisées, le premier mot ira dans var1, le second dans var2, et ainsi de suite. S’il y a moins de variables que de mots, tous les derniers mots vont dans la dernière variable.

```
$ cat read.sh

#!/bin/bash

echo "Continuer (O/N) ? \c"

read reponse

echo "reponse=$reponse"

case $reponse in

    O)

        echo "Oui, on continue"

        ;;

    N)

        echo "Non, on s’arrête"

        exit 0

        ;;

    *)

        echo "Erreur de saisie (O/N)"

        exit 1
```

;;

esac

echo "Vous avez continue. Tapez deux mots ou plus :\c"

read mot1 mot2

echo "mot1=$mot1\nmot2=$mot2"

exit 0

$ ./read.sh

Continuer (O/N) ? O

reponse=O

Oui, on continue

Vous avez continue. Tapez deux mots ou plus :salut les amis

mot1=salut

mot2=les amis

9. Les boucles

Elles permettent la répétition d’un bloc de commandes soit un nombre limité de fois, soit conditionnellement. Toutes les commandes à exécuter dans une boucle se placent entre les commandes **do** et **done**.

a. Boucle for

La boucle **for** ne se base pas sur une quelconque incrémentation de valeur mais sur une liste de valeurs, de fichiers ...

for var in liste

do

commandes à exécuter

done

La liste représente un certain nombre d’éléments qui seront successivement attribués à var.

Avec une variable

```
$ cat for1.sh

#!/bin/bash

for params in $@

do

        echo "$params"

done

$ ./for1.sh test1 test2 test3

test1

test2

test3
```

Liste implicite

Si vous ne précisez aucune liste à for, alors c’est la liste des paramètres qui est implicite. Ainsi le script précédent aurait pu ressembler à :

```
for params

do

        echo "$params"

done
```

Avec une liste d’éléments explicite

Chaque élément situé après le « in » sera utilisé pour chaque itération de la boucle, l’un après l’autre.

```
$ cat for2.sh

#!/usr/bin/sh

for params in liste liste2

do

        ls -l $params

done
```

```
$ ./for2.sh

-rw-r--r--   1 oracle   system      234 Aug 19 14:09 liste

-rw-r--r--   1 oracle   system      234 Aug 13 10:06 liste2
```

Avec des critères de recherche sur nom de fichiers

Si un ou plusieurs éléments de la liste correspond à un fichier ou à un motif de fichiers présents à la position actuelle de l'arborescence, la boucle for considère l'élément comme un nom de fichier.

```
$ cat for3.sh

#!/bin/bash

for params in *

do

        echo "$params \c"

        type_fic=`ls -ld $params | cut -c1`

        case $type_fic in

                -)      echo "Fichier normal" ;;

                d)      echo "Repertoire" ;;

                b)      echo "mode bloc" ;;

                l)      echo "lien symbolique" ;;

                c)      echo "mode caractere" ;;

                *)      echo "autre" ;;

        esac


done

$ ./for3.sh

case1.sh Fichier normal

dump.log Fichier normal
```

for1.sh Fichier normal

for2.sh Fichier normal

for3.sh Fichier normal

lien_fic1 lien symbolique

lien_fic2 lien symbolique

liste Fichier normal

liste1 Fichier normal

liste2 Fichier normal

param.sh Fichier normal

param2.sh Fichier normal

param3.sh Fichier normal

param4.sh Fichier normal

read.sh Fichier normal

rep1 Repertoire

resultat.txt Fichier normal

toto.txt Fichier normal

voir_a.sh Fichier normal

Avec une substitution de commande

Toute commande produisant une liste de valeurs peut être placée à la suite du « in » à l'aide d'une substitution de commande. La boucle for prendra le résultat de cette commande comme liste d'éléments sur laquelle boucler.

$ cat for4.sh

#!/bin/bash

echo "Liste des utilisateurs dans /etc/passwd"

for params in `cat /etc/passwd | cut -d: -f1`

do

echo "$params "

done

$ ./for4.sh

Liste des utilisateurs dans /etc/passwd

root

nobody

nobodyV

daemon

bin

uucp

uucpa

auth

cron

lp

tcb

adm

ris

carthic

ftp

stu

...

b. Boucle while

La commande **while** permet une boucle conditionnelle « tant que ». Tant que la condition est réalisée, le bloc de commande est exécuté. On sort si la condition n’est plus valable.

while condition

```
do
   commandes
done
```

ou :

```
while
bloc d'instructions formant la condition
do
   commandes
done
```

Par exemple :

```
$ cat while1.sh
#!/bin/bash
while
      echo "Chaine ? \c"
      read nom
      [ -z "$nom" ]
do
      echo "ERREUR : pas de saisie"
done
echo "Vous avez saisi : $nom"
```

Lecture d'un fichier ligne à ligne

```
#!/bin/bash
cat toto.txt | while read line
do
        echo "$line"
```

done

ou :

```
#!/bin/bash

while read line

do

        echo "$line"

done < toto.txt
```

Il y a une énorme différence entre les deux versions. Dans la première, notez la présence du tube (pipe) : la boucle est exécutée dans un second processus. Aussi toute variable modifiée au sein de cette boucle perd sa valeur en sortie !

c. Boucle until

La commande **until** permet une boucle conditionnelle « jusqu'à ». Dès que la condition est réalisée, on sort de la boucle.

```
until condition

do

   commandes

done
```

ou :

```
until

bloc d'instructions formant la condition

do

   commandes

done
```

d. true et false

La commande **true** ne fait rien d'autre que de renvoyer 0. La commande **false** renvoie toujours 1. De cette manière il est possible de réaliser des boucles sans fin. La seule manière de sortir de ces boucles est un exit ou un break.

Par convention, tout programme qui ne retourne pas d'erreur retourne 0, tandis que tout programme retournant une erreur, ou un résultat à interpréter, retourne autre chose que 0. C'est l'inverse en logique booléenne.

```
while true
do
   commandes
   exit / break
done
```

e. break et continue

La commande **break** permet d'interrompre une boucle. Dans ce cas le script continue après la commande **done**. Elle peut prendre un argument numérique indiquant le nombre de boucles à sauter, dans le cas de boucles imbriquées (rapidement illisible).

```
while true
do
   echo "Chaine ? \c"
   read a
   if [ -z "$a" ]
   then
      break
   fi
done
```

La commande **continue** permet de relancer une boucle et d'effectuer un nouveau passage. Elle peut prendre un argument numérique indiquant le nombre de boucles à relancer (on remonte de n boucles). Le script redémarre à la commande **do**.

f. Boucle select

La commande **select** permet de créer des menus simples, avec sélection par numéro. La saisie s'effectue au clavier avec le prompt de la variable PS3. Si la valeur saisie est incorrecte, une boucle s'effectue et le menu s'affiche à nouveau. Pour sortir d'un select il faut utiliser un **break**.

```
select variable in liste_contenu

do

   traitement

done
```

Si **in liste_contenu** n’est pas précisé, ce sont les paramètres de position qui sont utilisés et affichés.

```
$ cat select.sh

#!/bin/bash

PS3="Votre choix :"

echo "Quelle donnee  ?"

select reponse in Jules Romain Francois quitte

do

        if [[ "$reponse" = "quitte" ]]

        then

                break

        fi

        echo "Vous avez choisi $reponse"

done

echo "Au revoir."

exit 0


$ ./select.sh

Quelle donnee  ?

1) Jules

2) Romain
```

3) Francois

4) quitte

Votre choix :1

Vous avez choisi Jules

Votre choix :2

Vous avez choisi Romain

Votre choix :3

Vous avez choisi Francois

Votre choix :4

Au revoir.

10. Les fonctions

Les fonctions sont des bouts de scripts nommés, directement appelés par leur nom, pouvant accepter des paramètres et retourner des valeurs. Les noms de fonctions suivent les mêmes règles que les variables sauf qu’elles ne peuvent pas être exportées.

```
nom_fonction ()

{

   commandes

   return

}
```

Les fonctions peuvent être soit tapées dans votre script courant, soit dans un autre fichier pouvant être inclus dans l’environnement. Pour cela saisissez :

. nomfic

Le point suivi d’un nom de fichier charge son contenu (fonctions et variables) dans l’environnement courant.

La commande **return** permet d’affecter une valeur de retour à une fonction. Il ne faut surtout pas utiliser la commande **exit** pour sortir d’une fonction, sinon on quitte le script.

$ cat fonction

```
ll ()
{
	ls -l $@
}
 li ()
{
	ls -i $@
}
$ . fonction
$ li
  252 case1.sh     326 for4.sh      187 param.sh     897 resultat.txt
  568 dump.log     409 lien_fic1    272 param2.sh    991 toto.txt
  286 fonction     634 lien_fic2    260 param3.sh    716 voir_a.sh
  235 for1.sh     1020 liste         42 param4.sh   1008 while1.sh
  909 for2.sh      667 liste1       304 read.sh
  789 for3.sh     1006 liste2       481 rep1
```

11. Calculs et expressions

a. expr

La commande **expr** permet d’effectuer des calculs sur des valeurs numériques, des comparaisons, et de la recherche dans des chaînes de texte.

Opérateur	Rôle
+	Addition.
-	Soustraction. L’étoile étant reconnue par le shell comme un wildcard, il faut la verrouiller avec un antislash : *.
*	Multiplication.
/	Division.
%	Modulo.
!=	Différent. Affiche 1 si différent, 0 sinon.
=	Égal. Affiche 1 si égal, 0 sinon.

Opérateur	Rôle
<	Inférieur. Affiche 1 si inférieur, 0 sinon.
>	Supérieur. Affiche 1 si supérieur, 0 sinon.
<=	Inférieur ou égal. Affiche 1 si inférieur ou égal, 0 sinon.
>=	Supérieur ou égal. Affiche 1 si supérieur ou égal, 0 sinon.
:	Recherche dans une chaîne. Ex : expr Jules : J* retourne 1 car Jules commence par J. Syntaxe particulière : expr "Jules" : ".*" retourne la longueur de la chaîne.

```
$ expr 7 + 3

10

$ expr 7 \* 3

21

$ a=$(expr 13 - 10)

$ echo $a

3

$ cat expr1.sh

#!/bin/bash

cumul=0

compteur=0

nb_boucles=10

while [ "$compteur" -le "$nb_boucles" ]

do

    cumul=$(expr $cumul + $compteur)

    echo "$cumul=$cumul, boucle=$compteur"

    compteur=$(expr $compteur + 1)

done

$ ./expr1.sh

cumul=0, boucle=0
```

cumul=1, boucle=1

cumul=3, boucle=2

cumul=6, boucle=3

cumul=10, boucle=4

cumul=15, boucle=5

cumul=21, boucle=6

cumul=28, boucle=7

cumul=36, boucle=8

cumul=45, boucle=9

cumul=55, boucle=10

$ expr "Jules Cesar" : ".*"

11

b. Calculs avec bash

Le bash propose une forme simple de calculs sur les entiers, en plaçant l'opération entre **$((...))** :

$ a=1

$ a=$((a+1))

$ echo $a

2

$ b=2

$ a=$((a*b))

$ echo $a

4

Vous n'avez pas besoin de spécifier les $ des noms des variables entre les doubles parenthèses.

12. Une variable dans une autre variable

Voici un exemple :

```
$ a=Jules

$ b=a

$ echo $b

a
```

Comment afficher le contenu de a et pas simplement a ? En utilisant la commande **eval**. Cette commande située en début de ligne essaie d'interpréter, si possible, la valeur d'une variable précédée par deux « $ », comme étant une autre variable.

```
$ eval echo \$$b

Jules
```

```
$ cat eval.sh

cpt=1

for i in a b c

do

    eval $i=$cpt

    cpt=$((cpt+1))

done

echo $a $b $c
```

```
$ ./eval.sh

1 2 3
```

13. Traitement des signaux

La commande **trap** permet de modifier le comportement du script à la réception d'un signal.

commande	Réaction
trap '' signaux	Ignore les signaux. trap '' 2 3 ignore les signaux 2 et 3.
trap 'commandes' signaux	Pour chaque signal reçu exécution des commandes indiquées.

commande	Réaction
trap signaux	Restaure les actions par défaut pour ces signaux.

Dans l’exemple suivant trap empêche l’exécution du [Ctrl] **C** (SIGINT) et intercepte le signal SIGTERM :

```
$ cat trap.sh

!#/bin/bash



sortir()

{

        echo "Signal 15 recu"

        exit 0

}



trap ’’ 2

trap sortir 15



while true

do

        echo "Ctrl+C impossible!"

done



$ ./trap.sh

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!
```

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!

...

Depuis une autre console :

$ kill -2 12456 # aucun effet

$ kill -15 12456 # SIGTERM

Sur la premiere console :

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!

Ctrl+C impossible!

Signal 15 recu

14. Commande « : »

La commande **« : »** est généralement totalement inconnue des utilisateurs Unix. Elle retourne toujours la valeur 0 (réussite). Elle peut donc être utilisée pour remplacer la commande **true** dans une boucle par exemple :

while :

do

 ...

done

Cette commande placée en début de ligne, suivie d'une autre commande, traite la commande et ses arguments mais ne fait rien, et retourne toujours 0. Elle peut être utile pour tester les variables.

$: ls

$: ${X:?"Erreur"}

X : Erreur

TP : le shell et les commandes GNU :

Programmation Shell Niveau 1

Fonction Shell

THEME 111 Tâches administratives

111.1 Gérer les groupes, les utilisateurs et les répertoires personnels (4) et 111.2 Modifier les variables utilisateur et son environnement (3)

Administration des utilisateurs

1. Principe

a. Identification et authentification

L'**identification**, c'est savoir qui est qui, afin de déterminer les droits de la personne qui se connecte. Un utilisateur est identifié par un login.

L'**authentification**, c'est apporter la preuve de qui on est, par exemple via un secret partagé entre l'utilisateur et le système, et connus d'eux seuls. L'utilisateur est authentifié par un mot de passe.

b. Les utilisateurs

Un utilisateur est l'association d'un nom de connexion, le login, à un UID et au moins un GID.

UID : User ID.

GID : Group ID.

Les UID et les GID sont en principe uniques. Le login est unique. Il est cependant envisageable d'associer plusieurs logins au même UID, le système travaillant parfois avec le login.

L'UID identifie l'utilisateur (ou le compte applicatif) tout au long de sa connexion. Il est utilisé pour le contrôle de ses droits et de ceux des processus qu'il a lancé. Ce sont les UID et GID qui sont stockés au sein de la table des inodes, dans la table des processus, etc., et non les logins.

L'utilisateur dispose des attributs de base suivants :

un nom de connexion appelé le login ;

un mot de passe ;

un UID ;

un GID correspondant à son groupe principal ;

un descriptif ;

un répertoire de connexion ;

une commande de connexion ;

D’autres attributs sont disponibles via l’utilisation de la sécurité des mots de passe via shadow (voir la section concernée).

Les UID d’une valeur inférieure à 100 sont en principe associés à des comptes spéciaux avec des droits étendus. Ainsi l’UID de root, l’administrateur, est 0. Selon les distributions, à partir de 100, 500 ou 1000, et ce jusqu’à environ 60000, ce sont les UID des utilisateurs sans pouvoirs particuliers.

Un login a en principe une taille de 8 caractères. En fait Linux et d’autres systèmes acceptent une taille plus grande, mais avec la plupart des commandes l’affichage, voire la gestion des logins, est limité à 8 caractères.

Un login accepte la plupart des caractères. Il ne doit pas commencer par un chiffre. Il est possible de modifier la liste des caractères autorisés et de forcer la longueur et la complexité via les mécanismes d’authentification PAM et le fichier **/etc/login.defs**.

c. Les groupes

Chaque utilisateur fait partie d’au moins un groupe. Un groupe regroupe des utilisateurs. Comme pour les logins, le GID du groupe accompagne toujours l’utilisateur pour le contrôle de ses droits. Un utilisateur peut faire partie de plusieurs groupes, auquel cas il faut distinguer son groupe primaire des groupes secondaires.

Les groupes sont aussi des numéros. Il existe des groupes spécifiques pour la gestion de certaines propriétés du système et notamment l’accès à certains périphériques.

Le groupe primaire est celui qui est toujours appliqué à la création d’un fichier. Si l’utilisateur seb a pour groupe primaire users, alors les fichiers créés par seb auront comme groupe d’appartenance users.

Un utilisateur dispose de tous les droits associés à ses groupes secondaires. Si seb a comme groupe secondaire video et qu’un fichier dispose des droits d’écriture pour ce groupe, alors seb aura le droit de modifier son contenu.

La commande **id** permet de connaître les informations essentielles sur un utilisateur : uid, gid, groupes secondaires.

$ id seb

uid=1000(seb) gid=100(users) groupes=100(users),16(dialout),3(sys),

33(video)

Un fichier est créé par seb. Son propriétaire est seb et son groupe est le groupe principal de seb : users.

$ touch test

$ ls -l test

-rw-r--r-- 1 seb users 0 avr 10 14:30 test

d. Les mots de passe

Les mots de passe permettent d'authentifier les utilisateurs. Ils doivent être assez complexes pour ne pas être découverts facilement, mais assez intuitifs pour qu'ils s'en souviennent. Les mots de passe sont cryptés (MD5, DES par exemple) et ne sont pas directement lisibles sous leur forme cryptée par l'utilisateur afin que personne ne puisse tenter de le décrypter via un quelconque traitement.

Un utilisateur devrait changer régulièrement son mot de passe, ne jamais l'écrire quelque part ni le conserver sur lui. Vous verrez par la suite qu'il est possible de contraindre l'utilisateur à appliquer des règles de nommage et de durée de conservation.

Voici par exemple le résultat crypté par Blowfish (reconnaissable par le $2a$ commençant la chaîne) d'un mot de passe :

aher874oP47 vaut

$2a$10$CqutecUAlTGSFs2BPnVl..ntI8OEdy5j6gLI/cIKhHP4XZISdlGZO

2. Les fichiers

a. /etc/passwd

Le fichier **/etc/passwd** contient la liste des utilisateurs du système local. Il est lisible par tout le monde. Les informations qu'il contient sont publiques et utiles tant pour le système que pour les utilisateurs. Chaque ligne représente un utilisateur et est composée de sept champs.

Login:password:UID:GID:comment:homedir:shell

Champ 1 : le login ou nom d'utilisateur.

Champ 2 : sur les vieilles versions, le mot de passe crypté. Si un x est présent, le mot de passe est placé dans /etc/shadow. Si c'est un point d'exclamation le compte est verrouillé.

Champ 3 : le User ID.

Champ 4 : le GID, c'est-à-dire le groupe principal.

Champ 5 : un commentaire ou descriptif. C'est un champ d'information.

Champ 6 : le répertoire de travail, personnel, de l'utilisateur. C'est le répertoire dans lequel il arrive lorsqu'il se connecte.

Champ 7 : le shell par défaut de l'utilisateur. Mais ce peut être toute autre commande, y compris une commande interdisant la connexion.

b. /etc/group

Le fichier **/etc/group** contient la définition des groupes d'utilisateurs et pour chacun la liste des utilisateurs dont il est le groupe secondaire. Chaque ligne est composée de quatre champs :

Group:password:GID:user1,user2,…

Champ 1 : le nom du groupe.

Champ 2 : le mot de passe associé. Voyez l'explication juste en dessous.

Champ 3 : le Group Id.

Champ 4 : la liste des utilisateurs appartenant à ce groupe.

Il est inutile de replacer dans le quatrième champ les utilisateurs ayant ce groupe pour groupe principal, c'est induit.

Vous pouvez être surpris de voir la présence d'un champ de mot de passe pour les groupes. Dans la pratique il est très rarement utilisé. Comme il est bien entendu impossible de se connecter comme un groupe, l'explication est ailleurs. Un utilisateur a le droit de changer de groupe afin de prendre, temporairement tout du moins, un groupe secondaire comme groupe principal avec la commande **newgrp**.

Dans ce cas, l'administrateur peut mettre en place un mot de passe sur le groupe pour protéger l'accès à ce groupe en tant que groupe principal.

c. /etc/shadow

Le fichier **/etc/shadow** accompagne le fichier **/etc/passwd**. C'est là qu'est stocké, entre autres, le mot de passe crypté des utilisateurs. Pour être plus précis il contient toutes les informations sur le mot de passe et sa validité dans le temps. Chaque ligne est composée de 9 champs séparés par des : :

bean:$2a$10$AjADxPEfE5iUJcltzYA4wOZO.f2UZ0qP/8EnOFY.P.m10HifS7J8i:13

913:0:99999:7:::

Champ 1 : le login.

Champ 2 : le mot de passé crypté. Le xx initial indique le type de cryptage.

Champ 3 : nombre de jours depuis le 1er janvier 1970 du dernier changement de mot de passe.

Champ 4 : nombre de jours avant lesquels le mot de passe ne peut pas être changé (0 : il peut être changé n’importe quand).

Champ 5 : nombre de jours après lesquels le mot de passe doit être changé.

Champ 6 : nombre de jours avant l’expiration du mot de passe durant lesquels l’utilisateur doit être prévenu.

Champ 7 : nombre de jours après l’expiration du mot de passe après lesquels le compte est désactivé.

Champs 8 : nombre de jours depuis le 1er janvier 1970 à partir du moment où le compte a été désactivé.

Champ 9 : réservé.

Dans l’exemple de la ligne bean, le mot de passe a été changé 13913 jours après le 01/01/1970. Le mot de passe doit être changé avant 0 jours mais il est toujours valide car le champ suivant indique qu’il faut le changer au bout de 99999 jours (273 ans) et le champ 5 est vide (pas d’obligation de changement de mot de passe). Le compte est désactivé après 7 jours, ce qui évidemment ne risque pas d’arriver…



Pour connaître la date en fonction du 01/01/1970 utilisez la commande date comme ceci, en ajoutant le nombre de jours désiré :

$ date --date "1 jan 1970 +13984 days"

mar avr 15 00:00:00 CEST 2008

d. /etc/gshadow

Le fichier **/etc/gshadow** est le pendant du fichier précédent mais pour les groupes. Il n’est cependant pas supporté par défaut sur la plupart des distributions Linux et ne sera pas expliqué ici. Les mots de passe des groupes sont placés dans **/etc/group**.

3. Gestion des utilisateurs

a. Ajout

La création d’un utilisateur pourrait être entièrement effectuée à la main car Linux (et les autres Unix) s’appuient sur une suite de commandes qui ne font « que » modifier des fichiers plats déjà existants et qui créent et recopient des fichiers et dossiers au bon endroit avec les bons droits.

La création d'un utilisateur consiste à :

rajouter une ligne dans **/etc/passwd**,

rajouter d'une ligne dans **/etc/shadow**,

rajouter d'éventuelles informations dans **/etc/group**,

créer le répertoire personnel et mettre à jour son contenu avec **/etc/skel**,

changer les permissions et le propriétaire du répertoire personnel,

changer le mot de passe (encodé).

Vous pouvez créer directement un compte en éditant les fichiers avec un éditeur, bien que ce soit plutôt déconseillé. Si vous souhaitez le faire tout de même, utilisez la commande **vipw** qui va mettre à jour les divers caches associés à la gestion des comptes.

La commande **vipw** admet trois arguments :

-p : édition de **/etc/passwd**.

-g : édition de **/etc/group**.

-s : édition de **/etc/shadow**.

Tout ceci peut être effectué avec la commande **useradd**. Elle ajoute un nouveau compte et effectue les principales opérations :

création de l'utilisateur et remplissage des fichiers,

création d'un groupe privé d'utilisateur (de même nom que celui-ci),

création du répertoire personnel, remplissage et modification des droits.

useradd <options> login

Si aucune option n'est précisée, les valeurs par défaut sont récupérées au sein du fichier **/etc/defaults/useradd**.

Les options principales suivantes sont acceptées :

Option	Rôle
-m	Crée aussi le répertoire personnel. Elle est parfois comprise par défaut, mais il vaut mieux vérifier si le répertoire personnel est présent après l'utilisation de la commande si vous n'utilisez pas cette option.
-u	Précise l'UID numérique de l'utilisateur, pour le forcer. Autrement l'UID est calculé

Option	Rôle
	selon les règles du fichier login.defs et les UID existants.
-g	Précise le groupe principal de l’utilisateur, par GID ou par son nom (variable GROUP).
-G	Précise les groupes additionnels (secondaires, de l’utilisateur) séparés par des virgules (variable GROUPS).
-d	Chemin du répertoire personnel. Généralement /home/<login>, mais n’importe quel chemin peut être précisé (variable HOME/<login>).
-c	Un commentaire associé au compte. Il peut être quelconque mais est parfois utilisé par certaines commandes comme **finger**. Son contenu peut être modifié par l’utilisateur avec la commande **chfn**.
-k	Chemin du répertoire contenant le squelette de l’arborescence du répertoire utilisateur. C’est généralement /etc/skell (variable SKEL).
-s	Shell (commande de connexion) par défaut de l’utilisateur (variable SHELL). L’utilisateur peut le changer via la commande **chsh**.
-p	Le mot de passe de l’utilisateur. Attention ! le mot de passe doit déjà être crypté ! À moins de recopier le mot de passe d’un compte générique, vous préférerez utiliser ensuite la commande **passwd**.

La commande suivante crée le compte robert avec la plupart des options de base précisées. C’est juste un exemple car, sauf parfois le **-m**, si vous ne précisez rien ce sont les options par défaut par rapport à celles précisées dans le fichier **/etc/defaults/useradd**.

useradd -m -u 1010 -g users -G video,dialout,lp -s /bin/bash -d

/home/robert -c "Compte de Robert" robert

grep robert /etc/passwd

robert:x:1010:100:Compte de Robert:/home/robert:/bin/bash

La commande ne crée pas de mot de passe. Il faut le faire à la main avec la commande **passwd**.

passwd robert

Changing password for robert.

Nouveau mot de passe :

Retaper le nouveau mot de passe :

Mot de passe changé.

b. Sécurité des mots de passe

Changer de mot de passe

La commande **passwd** permet de gérer les mots de passe mais aussi les autorisations de connexion et la plupart des champs présents dans **/etc/shadow**.

Tout utilisateur a le droit de changer son mot de passe, dans le délai précisé par le champ 4 de /etc/shadow. L'action par défaut est de changer le mot de passe de l'utilisateur courant. L'ancien mot de passe est demandé par sécurité (notamment pour empêcher une personne mal intentionnée de modifier votre mot de passe derrière votre dos). La saisie est masquée.

$ id

uid=1000(seb) gid=100(users) …

$ passwd

Changing password for seb.

Ancien mot de passe :

Nouveau mot de passe :

Retaper le nouveau mot de passe :

Mot de passe changé.

Les modules **PAM** (**Pluggable Authentification Module**) peuvent imposer des contraintes plus ou moins sévères pour le choix du mot de passe : de telle longueur, pas basé sur un mot du dictionnaire, etc. Voyez ce qu'il se passe en tentant d'utiliser successivement toto (trop court), azerty (trop long) et Martine (dictionnaire) :

$ passwd

Changing password for seb.

Ancien mot de passe :

Nouveau mot de passe :

Mot de passe incorrect : trop court

Nouveau mot de passe :

Mot de passe incorrect : trop simple

Nouveau mot de passe :

Mot de passe incorrect : basé sur un mot du dictionnaire

passwd: Nombre maximum de tentatives épuisées pour le service

L'utilisateur root a le droit de modifier les mots de passe de tous les utilisateurs du système, sans avoir à connaître le précédent mot de passe. Mieux : il peut forcer l'utilisation d'un mot de passe même si celui-ci n'est pas validé par PAM :

passwd seb

Changing password for seb.

Nouveau mot de passe :

Mot de passe incorrect : basé sur un mot du dictionnaire

Retaper le nouveau mot de passe :

Mot de passe changé.

Gérer les informations de validité

Tous les champs de **/etc/shadow** peuvent être modifiés par la commande **passwd**. Voici quelques options disponibles.

Option	Rôle
-l	Lock : verrouille le compte en rajoutant un ! devant le mot de passe crypté.
-u	Unlock : déverrouille le compte. Il n'est pas possible de déverrouiller un compte qui n'a pas de mot de passe, il faut utiliser en plus -f pour cela.
-d	(root) Supprime le mot de passe du compte.
-n <j>	(root) Durée de vie minimale en jours du mot de passe.
-x <j>	(root) Durée de vie maximale en jours du mot de passe.
-w <j>	(root) Nombre de jours avant avertissement.
-i <j>	(root) Délai de grâce avant désactivation si le mot de passe est expiré.
-S	(root) Statut du compte.

Dans l'exemple suivant le compte bean est modifié comme ceci :

Il doit attendre 5 jours après saisie d'un nouveau mot de passe pour pouvoir le changer,

Son mot de passe est valide 45 jours,

Il est prévenu 7 jours avant qu'il doit changer de mot de passe,

S'il ne change pas de mot de passe après 45 jours, il dispose encore de 5 jours avant d'être désactivé.

passwd -n 5 -x 45 -w 7 -i 5 bean

Password expiry information changed.

Voici la ligne de **/etc/shadow** associée.

bean:$2a$10$dwbUGrC75bs3l52V5DHxZefkZyB6VTHsLH5ndjsNe/vF/HAzHOcR2:13

984:5:45:7:5::

La commande **chage** permet de faire à peu près la même chose. Elle n'est accessible que par root. Lancée sans autre argument que le login de l'utilisateur, elle est interactive. Notez à la fin la possibilité de modifier la date du dernier changement du mot de passe et une date fixe d'expiration du mot de passe (champ 8) :

chage bean

Changing aging information for bean.

Minimum Password Age [7]:

Maximum Password Age [40]:

Password Expiration Warning [10]:

Password Inactive [5]:

Last Password Change (YYYY-MM-DD) [2008-04-10]:

Account Expiration Date (YYYY-MM-DD) [1969-12-31]: 2010-01-01

Aging information changed.

Voici la ligne /etc/shadow résultante :

bean:$2a$10$dwbUGrC75bs3l52V5DHxZefkZyB6VTHsLH5ndjsNe/vF/HAzHOcR2:13

979:7:40:10:5:14610:

Les paramètres suivants sont acceptés :

Option	Rôle
-m	Mindays : équivaut à passwd -n.
-M	Maxdays : équivaut à passwd -x.
-d	Date de dernière modification du mot de passe (depuis le 01/01/1970).
-E	Date d'expiration du mot de passe (depuis le 01/01/1970).
-I	Inactive : équivaut à passwd -i.
-W	Warndays : équivaut à passwd -w.
-l	List : affiche tous les détails.





Les détails sont bien plus lisibles avec chage qu'avec passwd :

passwd -S bean

bean PS 04/10/2008 7 40 10 5

chage -l bean

Minimum: 7

Maximum: 40

Warning: 10

Inactive: 5

Last Change: avr 10, 2008

Password Expires: mai 20, 2008

Password Inactive: mai 25, 2008

Account Expires: jan 01, 2010



Un utilisateur quelconque peut afficher ses propres détails, mais son mot de passe lui sera demandé.

c. Modification

Utilisez la commande **usermod** pour modifier un compte. Elle prend la même syntaxe et les mêmes options que useradd mais dispose aussi d'une syntaxe complémentaire qui nécessite quelques précisions.

Option	Rôle
-L	Lock du compte, comme passwd -l.
-U	Unlock du compte, comme passwd -u.
-e <n>	Expire : le mot de passe expire n jours après le 01/01/1970.
-u <UID>	Modifie l'UID associé au login. Le propriétaire des fichiers appartenant à l'ancien UID au sein du répertoire personnel est modifié en conséquence.
-l <login>	Modifie le nom de login.
-m	Move : implique la présence de -d pour préciser un nouveau répertoire personnel. Le contenu de l'ancien répertoire est déplacé dans le nouveau.

d. Suppression

Supprimez un utilisateur avec la commande **userdel**. Par défaut le répertoire personnel n'est pas supprimé. Vous devez pour ceci passer l'option -r.

userdel -r bean

4. Gestion des groupes

a. Ajout

Vous pouvez créer un groupe directement dans le fichier **/etc/group** ou bien passer par les commandes associées. Si vous éditez le fichier à la main, utilisez la commande **vigr** (ou vipw -g).

La commande **groupadd** permet de créer un groupe. Sa syntaxe simple accepte l'argument **-g** pour préciser un GID précis.

grep amis /etc/group

amis:!:1234:

b. Modification

La commande **groupmod** permet de modifier un groupe. Ses paramètres sont les suivants :

Option	Rôle
-n <nom>	Renomme le groupe.
-g <GID>	Modifie le GID. Attention, le groupe d'appartenance des fichiers concernés n'est pas modifié.
-A <user>	Ajoute l'utilisateur spécifié dans le groupe (groupe secondaire).
-R <user>	Supprime l'utilisateur spécifié du groupe.

groupmod -R seb amis

grep amis /etc/group

amis:!:1234:

c. Suppression

La commande **groupdel** supprime un groupe. La commande vérifie d'abord si le groupe que vous voulez supprimer est le groupe principal d'un utilisateur. Dans ce cas le groupe ne peut pas être supprimé.

Par contre aucune action autre que celle consistant à supprimer la ligne correspondant dans /etc/group n'est effectuée : c'est à vous de vérifier le système de fichiers (et la configuration des applications si besoin) pour supprimer toute trace de ce groupe.

groupdel amis

5. Commandes additionnelles

a. Conversion des fichiers

Vous utiliserez probablement rarement les commandes suivantes. Elles sont surtout utiles dans le cadre de migrations de serveurs Linux vers d'autres Unix, et vice versa. Quelques systèmes Unix n'utilisent pas par défaut (il faut l'activer après) la gestion des comptes avec les fichiers **shadow**. Dans ce cas il peut être nécessaire de convertir les fichiers **/etc/shadow** et **/etc/passwd** en un seul et unique **/etc/passwd**. C'est le rôle de la commande **pwunconv**.

Dans l'exemple suivant, le fichier **/etc/passwd** est converti. Une fois la commande exécutée, toute trace de **/etc/shadow** a disparu.

pwunconv

grep bean /etc/passwd

bean:$2a$10$dwbUGrC75bs3l52V5DHxZefkZyB6VTHsLH5ndjsNe/vF/HAzHOcR2:1001:100:toto:/home/bean:/bin/bash

ls -l /etc/shadow

ls: ne peut accéder /etc/shadow: Aucun fichier ou répertoire de ce type



Attention : la commande **pwunconv** est destructive. Toutes les informations des durées de validité des mots de passe sont détruites. Il n'est plus possible d'utiliser les options diverses de chage ou de passwd concernant les durées de validité.

La commande **pwconv** fait l'inverse : elle crée le fichier **/etc/shadow** associé à **/etc/passwd**, y déplace les mots de passe et y place les réglages par défaut tels que définis dans le fichier **/etc/login.defs**.

grep bean /etc/passwd

bean:x:1001:100:toto:/home/bean:/bin/bash

p64p17bicb3:/home/seb # grep bean /etc/shadow

bean:$2a$10$dwbUGrC75bs3l52V5DHxZefkZyB6VTHsLH5ndjsNe/vF/HAzHOcR2:13

984:0:99999:7:::0

Les commandes **grpconv** et **grpunconv** font la même chose pour les groupes, mais ici sans succès puisque la plupart des distributions ne supporte pas les groupes en shadow.

b. Vérifier la cohérence

Il peut être utile de lancer des outils de vérification de la cohérence des fichiers des groupes et des mots de passe. Si vous avez l'habitude de modifier ces fichiers à la main, rien ne garantit que tout est correct : un groupe peut être manquant, un shell inexistant, un répertoire personnel absent, etc. La commande **pwck** effectue une vérification des fichiers **/etc/passwd** et **/etc/shadow** et reporte les erreurs.

Voici un exemple où les données de l'utilisateur bean ont été volontairement altérées pour tester la commande : le groupe n'existe pas, et le shell non plus. Sur ces entrefaites un autre problème a été découvert sur la machine de test.

pwck

Checking `/etc/passwd'

User `suse-ncc': directory `/var/lib/YaST2/suse-ncc-fakehome' does

not exist.

User `bean': unknown group `14400'

User `bean': shell `/bin/bashr' is not executable.

Checking `/etc/shadow'.

La commande **grpck** fait la même chose pour les groupes. Dans ce cas les contrôles sont moins étendus, se limitant aux doublons et à l'existence des utilisateurs pour les groupes secondaires.

grpck

Checking `/etc/group'

Group `users': unknown user `zorg'

c. Vérifier les connexions

Vous pouvez tracer les connexions sur votre machine à l'aide de deux commandes. La commande **lastlog** se base sur le contenu de /var/log/lastlog. Elle accepte les paramètres **-u** (précision d'un utilisateur) et **-t** pour rechercher les connexions des n derniers jours.

$ lastlog -u seb

Username Port Latest

seb pts/4 jeu avr 10 15:13:46 +0200 2008

$ lastlog -t 10

Username Port Latest

root pts/3 jeu avr 10 14:48:13 +0200 2008

seb pts/4 jeu avr 10 15:13:46 +0200 2008

La commande **last** fait à peu près la même chose, mais se base sur /var/log/wtmp qui fournit des informations supplémentaires comme l'origine de la connexion (IP, nom de la console, etc.) et les dates de connexion et de déconnexion, ainsi que la durée de connexion et si l'utilisateur est encore connecté.

$ last

seb pts/1 Tue Apr 15 14:39 still logged in

seb pts/4 Tue Apr 15 12:06 still logged in

seb pts/6 Tue Apr 15 10:07 still logged in

seb pts/3 Mon Apr 14 13:36 - 15:18 (1+01:42)

seb pts/1 Thu Apr 10 15:39 - 11:58 (4+20:18)

seb pts/4 localhost Thu Apr 10 15:13 - 15:39 (00:25)

seb pts/4 localhost Thu Apr 10 15:13 - 15:13 (00:00)

seb pts/4 localhost Thu Apr 10 15:12 - 15:13 (00:00)

...

d. Actions de l'utilisateur

L'utilisateur dispose de certaines actions sur les informations de son compte. Il peut notamment :

changer son shell de connexion,

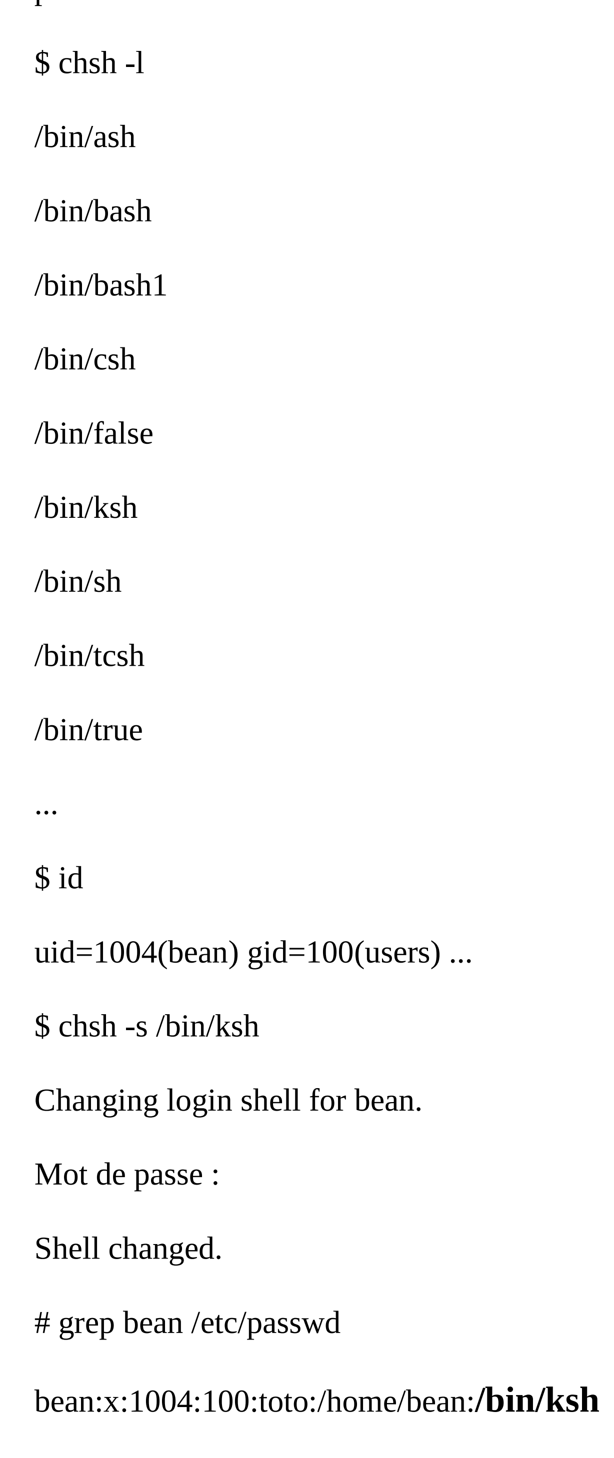
changer ses informations personnelles,

changer de groupe principal,

prendre l'identité de quelqu'un d'autre.

Changer de shell

La commande **chsh** permet à l'utilisateur de modifier définitivement (ou jusqu'à la prochaine commande chsh) de shell de connexion. Il ne peut pas choisir n'importe quoi. Le shell (ou toute autre commande) doit être présent dans **/etc/shells**. Cette liste est accessible via le paramètre **-l** de la commande. La modification est faite au sein de **/etc/passwd**. Seul root a le droit de le modifier pour d'autres utilisateurs. Le nouveau shell est précisé avec le paramètre **-s**.



$ chsh -l

/bin/ash

/bin/bash

/bin/bash1

/bin/csh

/bin/false

/bin/ksh

/bin/sh

/bin/tcsh

/bin/true

...

$ id

uid=1004(bean) gid=100(users) ...

$ chsh -s /bin/ksh

Changing login shell for bean.

Mot de passe :

Shell changed.

grep bean /etc/passwd

bean:x:1004:100:toto:/home/bean:**/bin/ksh**

Changer le commentaire

Le commentaire du fichier **/etc/passwd** peut être modifié par l'utilisateur à l'aide de la commande **chfn**. Il est préférable de l'utiliser de manière interactive (le passage de paramètre en mode non interactif est réservé à root).

$ chfn

Changing finger information for bean.

Mot de passe :

Enter the new value, or press ENTER for the default

Full Name: toto

Room Number []: Mister Bean

Work Phone []: 0102030405

Home Phone []: 0605040302

Other:

Finger information changed.

$ grep bean /etc/passwd

bean:x:1004:100**:Mister Bean,0102030405,0605040302:**/home/bean:/bin/bash

Changer de groupe principal

La commande **newgrp** permet de changer à titre temporaire de groupe principal, à condition que le nouveau groupe précisé soit un groupe secondaire de l'utilisateur et/ou que l'utilisateur dispose du mot de passe du groupe. Utilisée seule, **newgrp** revient au groupe d'origine. Les modifications sont temporaires, le fichier des mots de passe n'est pas modifié.

$ id

uid=1004(bean) gid=100(users) groupes=16(dialout),33(video),100(users)

$ newgrp video

$ id

uid=1004(bean) **gid=33(video)** groupes=16(dialout),33(video),100(users)

Que se passe-t-il si vous tentez de prendre comme groupe principal un groupe ne faisant pas partie de vos groupes secondaires mais qui est protégé par un mot de passe ? Dans l'exemple suivant, bean tente de prendre comme groupe principal grptest.

$ id

uid=1004(bean) gid=100(users) groupes=16(dialout),33(video),100(users)

$ newgrp grptest

Password:

$ id

uid=1004(bean) **gid=1000(grptest)** groupes=16(dialout),33(video),

100(users),1000(grptest)

Changer d'identité

L'utilisateur peut endosser, le temps d'une commande ou de toute une session, l'identité d'une autre personne. Il s'agit généralement de root, car vous savez qu'il ne faut jamais (ou au moins éviter) de se connecter en permanence en tant que root. Donc pour les tâches administratives il faut pouvoir devenir root (ou un autre utilisateur) le temps nécessaire.

La commande **su** (**substitute user**) permet d'ouvrir une session, ou d'exécuter un shell ou une commande donnée avec une autre identité. Évidemment, vous devez connaître le mot de passe de cet utilisateur.

su [-c commande] [-s shell] [-] [utilisateur]

Si aucun utilisateur n'est précisé, c'est root qui est utilisé.

$ id

uid=1000(seb) gid=100(users) …

seb@p64p17bicb3:~> su

Mot de passe :

p64p17bicb3:/home/seb # id

uid=0(root) gid=0(root) groupes=0(root)

Notez que c'est l'environnement de l'utilisateur d'origine qui est utilisé par défaut. L'environnement de root (ou de tout autre utilisateur) n'est pas chargé :

echo $LOGNAME $USER

seb seb

Pour charger l'environnement complet de l'utilisateur cible, rajoutez le tiret en paramètre :

$ su - bean

Mot de passe :

$ echo $USER $LOGNAME

bean bean

Pour exécuter ponctuellement une commande avec une autre identité, utilisez **-c**.

$ su -c "make install"

La commande **sg** (**subsitute group**) est identique à **su** mais elle prend un nom de groupe en argument.

6. Configuration avancée



Les descriptions des fichiers qui suivent dépendent fortement des versions des commandes, de leurs options de compilation et de la politique de sécurité appliquée par les éditeurs des diverses distributions. Il est possible que certains fichiers ne soient pas présents, et que d'autres contiennent des paramètres différents.

a. /etc/default/useradd

Le fichier **/etc/default/useradd** contient un certain nombre de variables définissant les règles par défaut à appliquer à la création d'un utilisateur :

son groupe,

la racine de son répertoire personnel (là où celui-ci sera situé),

s'il est actif ou non,

le shell,

son ou ses groupes secondaires,

l'endroit où est situé le squelette des comptes (structure de base d'un répertoire utilisateur),

la création ou non d'un spool (dépôt) de courrier,

etc.

$ cat /etc/defaults/useradd

GROUP=100

HOME=/home

INACTIVE=-1

EXPIRE=

SHELL=/bin/bash

SKEL=/etc/skel

GROUPS=video,dialout

CREATE_MAIL_SPOOL=no

b. /etc/default/passwd

Le fichier **/etc/default/passwd** contient quelques règles utilisées par la commande **passwd** pour le cryptage des mots de passe. Il est possible de définir des règles de cryptage globales, mais aussi par type de fichier, et de passer quelques options selon la méthode.

$ cat /etc/default/passwd

Cryptage par defaut

CRYPT=md5

Cryptage pour les fichiers (/etc/shadow)

CRYPT_FILES=blowfish

option pour blowfish

BLOWFISH_CRYPT_FILES=10

Pour NIS

CRYPT_YP=des

c. /etc/default/su

Le fichier **/etc/default/su** permet de configurer le fonctionnement de la commande **su**. Par défaut **su** avec le paramètre **-** met en place un nouveau PATH car il charge l'environnement

de l'utilisateur ciblé. Vous pouvez modifier ceci et mettre en place votre propre PATH, ou conserver l'ancien.

Change le PATH meme sans le tiret

ALWAYS_SET_PATH=no

Path par défaut

PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin

Path par défaut pour root

SUPATH=/usr/sbin:/bin:/usr/bin:/sbin:/usr/X11R6/bin

d. /etc/login.defs

Le fichier **/etc/login.defs** est utilisé par de nombreuses commandes comme **login**, **useradd**, **groupadd**, **passwd** pour définir quelques valeurs par défaut et la validité des logins. Son contenu peut varier suivant les distributions. Il peut contenir :

une règle de validité des comptes (caractères autorisés, longueur, etc.),

les UID min et max lors de la création d'un utilisateur,

les GID min et max lors de la création d'un groupe,

les commandes à appeler pour les ajouts/modifications/créations d'utilisateur,

les règles par défaut pour la validité des mots de passe,

la création ou non d'un répertoire personnel,

etc.



Le contenu du fichier login.defs peut être en conflit avec certaines options des autres fichiers présents dans /etc/default. Dans ce cas, vous devez vérifier dans les manuels (ou manpages) de votre distribution quel fichier prévaut.

Login autorisé meme si l'accès au home est impossible

DEFAULT_HOME   yes

Path par défaut pour la commande login

ENV_PATH /usr/local/bin:/usr/bin:/bin

Idem pour la connexion root

ENV_ROOTPATH /sbin:/bin:/usr/sbin:/usr/bin

Délai en secondes entre deux tentatives de login

FAIL_DELAY 3

Les utilisateurs dans le fichier ne voient pas les messages de cnx

HUSHLOGIN_FILE /etc/hushlogins

Affiche la date de dernière connexion

LASTLOG_ENAB yes

Les tentatives de connexion des logins inexistants sont tracées

LOG_UNKFAIL_ENAB no

trois tentatives de login en cas de mauvais mot de masse

LOGIN_RETRIES 3

Timeout de 60 secondes

LOGIN_TIMEOUT 60

Emplacement du motd (ou des , séparés par des :)

MOTD_FILE /etc/motd

type des terminaux par défaut

TTYTYPE_FILE /etc/ttytype

Permission par défaut des terminaux

TTYGROUP tty

TTYPERM 0620

Réclame ou non un mot de passe pour chsh et chfn

CHFN_AUTH yes

Restriction du chfn (f:nom, r:bureau, w:tel travail, h:tel maison)

CHFN_RESTRICT rwh

PASS_MAX_DAYS Duree max du mot de passe

PASS_MIN_DAYS Délai min entre deux changements de mot de passe

PASS_WARN_AGE Avertissement avant le changement

PASS_MAX_DAYS 99999

PASS_MIN_DAYS 0

PASS_WARN_AGE 7

UID/GID Min et Max par défaut et systeme pour useradd

SYSTEM_UID_MIN 100

SYSTEM_UID_MAX 499

UID_MIN 1000

UID_MAX 60000

#

Min/max values for automatic gid selection in groupadd

#

SYSTEM_GID_MIN to SYSTEM_GID_MAX inclusive is the range for

GIDs for dynamically allocated administrative and system groups.

GID_MIN to GID_MAX inclusive is the range of GIDs of dynamically

allocated groups.

#

SYSTEM_GID_MIN 100

SYSTEM_GID_MAX 499

GID_MIN 1000

GID_MAX 60000

Regexp pour les noms de logins autorisés via login/useradd

CHARACTER_CLASS [A-Za-z_][A-Za-z0-9_.-]*[A-Za-z0-9_.$-]\?

Umask par défaut pour la création du homedir

UMASK 022

Commande réellement exécutée lors de l'ajout d'un groupe

GROUPADD_CMD /usr/sbin/groupadd.local

Commande réellement exécutée lors de l'ajout d'un utilisateur

USERADD_CMD /usr/sbin/useradd.local

Commande exécutée avant la suppression d'un groupe

USERDEL_PRECMD /usr/sbin/userdel-pre.local

Commande exécutée après la suppression d'un groupe

USERDEL_POSTCMD /usr/sbin/userdel-post.local

En plus des commandes **useradd** et **userdel**, voici une liste non exhaustive des commandes qui utilisent les paramètres de ce fichier :

login : DEFAULT_HOME, ENV_PATH, ENV_ROOTPATH, FAIL_DELAY, HUSHLOGIN_FILE, LASTLOG_ENAB, LOG_UNKFAIL_ENAB, LOGIN_RETRIES, LOGIN_TIMEOUT, MOTD_FILE, TTYPERM, TTYTYPE_FILE ;

newusers : PASS_MAX_DAYS, PASS_MIN_DAYS, PASS_WARN_AGE, UMASK ;

passwd : OBSCURE_CHECKS_ENAB, PASS_MAX_LEN, PASS_MIN_LEN, PASS_ALWAYS_WARN, CRACKLIB_DICTPATH, PASS_CHANGE_TRIES ;

pwconv : PASS_MAX_DAYS, PASS_MIN_DAYS, PASS_WARN_AGE.

7. Notifications à l'utilisateur

a. /etc/issue

Lorsqu'un utilisateur se connecte depuis la console, un message est généralement affiché juste avant l'invite de saisie de son login. Ce message est contenu dans le fichier **/etc/issue**. C'est un message d'accueil et à ce titre il peut contenir tout ce que vous voulez. Par défaut, il contient généralement le nom de la distribution Linux et le numéro de version du noyau.

Voici l'exemple du contenu de **/etc/issue** sur une distribution Mandriva :

$ cat issue

Mandriva Linux release 2008.1 (Official) for i586

Kernel 2.6.24.4-desktop-1mnb on an i686 / \l

b. /etc/issue.net

Le message d'accueil peut être différent lorsqu'un utilisateur se connecte depuis une console distante (telnet, ssh, etc.). C'est souvent le même mais sans les caractères de contrôles liés à un shell donné. Pour modifier ce message spécifique, éditez le contenu du fichier **/etc/issue.net**.

c. /etc/motd

Motd signifie Message of the day, le message du jour. Une fois l'utilisateur connecté depuis une console (locale ou distante), un message peut être affiché. L'administrateur peut modifier ce message en éditant le fichier **/etc/motd**. Par défaut il est vide. Vous pouvez par exemple modifier ce fichier pour prévenir vos utilisateurs qu'un reboot de maintenance aura lieu tel jour à telle heure, ceci évitant d'envoyer n mails…

8. Aperçu de PAM

PAM (**Pluggable Authentification Modules**) est un ensemble de modules et une bibliothèque permettant de mettre en place des mécanismes d'authentification avancés pour tous les outils nécessitant une sécurité accrue. L'authentification est basée sur des modules. Chaque module peut utiliser des mécanismes différents pour tenter d'authentifier un utilisateur. L'un se basera sur une authentification Unix classique, un autre sur LDAP, un troisième sur Active Directory, et un dernier sur la reconnaissance d'une emprunte digitale. Ce mécanisme permet aussi la vérification via un dongle USB...

Le principe est assez simple, la configuration plus compliquée. Un outil appelle la bibliothèque **libpam.so** pour un besoin d'authentification. La bibliothèque lit un fichier de configuration où l'appel à plusieurs modules peut être demandé. Chaque module appelé retourne vrai ou faux. Suivant la configuration, vrai peut être un pré-requis pour continuer avec un autre module ou être suffisant pour se connecter.

Les fichiers de configuration sont placés dans **/etc/pam.d**. Il en existe généralement un par application utilisant PAM et il porte le même nom. Si le fichier est manquant c'est « **other** » qui est utilisé. La syntaxe est la suivante.

Type_module contrôle module arguments

Type_module

auth : module d'authentification (par exemple demande de login et de mot de passe).

account : autorisation, gestion de comptes (vérification de l'utilisateur pour le service donné. Est-il autorisé ?).

password : vérification et mise à jour des informations de sécurité (ex : le mot de passe est-il encore valable et si non, demande d'un nouveau mot de passe).

session : modification de l'environnement de l'utilisateur.

contrôle

required : réussite requise. En cas d'échec, les modules restants sont tout de même appelés mais quoi qu'il arrive au final PAM retournera un échec.

requisite : un échec termine immédiatement l'authentification. La réussite l'autorise à continuer.

sufficient : une réussite contourne les autres modules. Autrement dit PAM retourne ok quoi qu'il arrive en cas de réussite ici.

optional : le résultat est ignoré.

module

pam_unix.so : authentification standard via la fonction C getpw().

pam_env.so : définition des variables d'environnement.

pam_securetty.so : interdit une connexion super-utilisateur (root) depuis un terminal non sécurisé. La liste des terminaux autorisés est placée dans /etc/securetty.

pam_stack.so : appelle un autre service PAM pour le chargement de modules supplémentaires.

pam_nologin.so : interdit la connexion d'utilisateurs si le fichier **/etc/nologin** est présent. Dans ce cas son contenu est affiché.

pam_deny.so : retourne toujours un échec.

pam_console.so : donne des permissions supplémentaires à un utilisateur local.

Les paramètres dépendent de chaque module. Voici un exemple dans le cas classique d'une ouverture de session par la console. Dans ce cas, le shell de connexion appelle la commande **login**. Attention ce n'est qu'un exemple !

```
$ cat login

#%PAM-1.0

auth      requisite    /lib/security/pam_securetty.so

auth      required     /lib/security/pam_env.so

auth      sufficient   /lib/security/pam_unix.so

auth       required    /lib/security/pam_deny
```

auth required /lib/security/pam_nologin.so

La première ligne vérifie si root tente de se connecter depuis un terminal non sécurisé (ex : telnet, rlogin, rsh, etc.). Si c'est le cas, PAM retourne directement faux et l'authentification échoue directement.

La seconde ligne charge l'environnement de l'utilisateur. Un échec ici n'empêche pas l'exécution des lignes suivantes, mais au bout du compte PAM retournera faux.

La troisième ligne tente une authentification via les mécanismes Unix classiques (fichier des mots de passe, NIS, etc.). La réussite ici stoppe la procédure : l'utilisateur est directement authentifié. Autrement, on passe à la ligne suivante.

La quatrième ligne retourne toujours faux. Les modules d'authentification précédents ayant échoué, la connexion de l'utilisateur échoue. La ligne suivante est quand même exécutée.

Si le fichier **/etc/nologin** existe, il est affiché.

Il est possible d'interdire l'accès à une liste d'utilisateurs donnés. Placez les noms des utilisateurs interdits dans /etc/nologinusers et ajoutez la ligne suivante dans le fichier de configuration PAM. Dans notre exemple, il faudrait l'ajouter entre la deuxième et la troisième ligne.

auth required /lib/security/pam_listfile.so onerr=suceed item=user

sense=deny file=/etc/nologinusers

La distribution Red Hat est configurée de telle manière que le fichier **/etc/security/system-auth** est appelé dans toutes les configurations PAM ou presque. Exemple du fichier de configuration login :

auth required /lib/security/pam_securetty.so

auth required /lib/security/pam_stack.so service=system-auth

auth required /lib/security/pam_nologin.so

...

Dans ce cas, vous auriez mis notre ligne en dessous de celle de pam_nologin.so. Cependant il faut garder à l'esprit que le fichier de configuration **/etc/pam.d/login** n'est utilisé que pour l'authentification depuis la commande **login** et donc une connexion depuis un terminal (console texte). Depuis une fenêtre de connexion graphique (X-Window, x/k/gdm) vous n'aurez pas l'effet souhaité. Dans ce cas, la modification aurait dû être effectuée dans **/etc/pam.d/system-auth**.

auth required /lib/security/pam_env.so

auth sufficient /lib/security/pam_unix.so likeauth nullok

```
auth        required      /lib/security/pam_deny.so

...
```

Attention encore une fois à placer la ligne au bon endroit. Après ce bloc, la ligne n'aura aucun effet à cause de la ligne **sufficient** si l'utilisateur a un login et un mot de passe valides. Vous placerez donc la ligne au début des lignes **auth**.

111.2 Modifier les variables utilisateur et son environnement (3)

Voire chapître 109.1 Modifier l'environnement du shell

111.3 Configurer syslog (3)

Les traces (logs) du système

1. Principe

Lorsque le système démarre, fonctionne et effectue tout type d'action, ses actions et celles de la plupart de ses services sont tracées dans divers fichiers. Deux services sont spécialisés dans la réception des messages à écrire dans ces fichiers :

klogd : **kernel log daemon**, chargé de la gestion des informations émises par le noyau.

syslogd : **system log daemon**, chargé de la gestion des informations émises par tout type de service et éventuellement le noyau.



Certaines distributions utilisent maintenant **syslog-ng** dont les règles de traitement des messages, basées sur des expressions régulières, ont fortement évolué. Le principe reste cependant exactement le même.

Historiquement le service syslogd gérait aussi les messages émis par le noyau. Il en est toujours capable, mais la quantité de messages émis, les différents niveaux de sévérité et les nouvelles méthodes d'accès aux messages du noyau font qu'il a semblé important et pertinent de séparer la gestion des messages du noyau de ceux émis par les services.

Les messages importants émis par un composant du système devraient passer par le service syslogd. Ceci n'empêche pas, au contraire, qu'un service puisse gérer ses propres traces dans ses propres fichiers. Les traces applicatives ne devraient pas être placées dans les traces de système. Les traces d'accès aux pages Web d'un serveur Apache n'ont rien à y faire. Par

contre les traces de connexion au système (via la console, ssh, telnet, etc.) ont un intérêt important et doivent être présentes dans les fichiers logs du système.

Dans la suite de l'ouvrage, les traces seront appelées par leur nom d'usage courant : les **logs**.

2. Les messages

Le service **klogd** gère les messages émis par le noyau. Il dispose de deux sources d'accès aux messages :

le système de fichiers virtuel /proc, utilisé par défaut s'il est présent, et notamment /proc/kmsg ;

les appels systèmes via l'API du noyau, notamment sys_syslog, si /proc est absent ou si le paramètre **-s** a été passé à klogd.

Les messages du noyau ont des niveaux de priorité différents, étagés de 0 (haute priorité) à 7 (message de débogage) :

Niveau	Alias système	Signification
0	**EMERG**	Le système est inutilisable.
1	**ALERT**	Une action doit être prise immédiatement.
2	**CRIT**	Problème critique.
3	**ERR**	Erreur.
4	**WARNING**	Avertissement.
5	**NOTICE**	Normal mais nécessite une attention particulière.
6	**INFO**	Information standard.
7	**DEBUG**	Trace de débogage du noyau.

Le service klogd renvoie les messages de niveau 0 à 6 à syslogd qui redirigera ceux-ci dans les fichiers de logs et éventuellement sur les consoles concernées. Les informations de débogage de niveau 7 ne sont pas tracées par défaut.

Le service **syslogd** (ou syslog-ng) reçoit les messages issus des services mais aussi de klogd. Il les dispatche ensuite selon l'émetteur, la sévérité, dans des fichiers, des consoles, sous forme de mails aux utilisateurs du système (root par exemple), etc.

Les actions les plus courantes sont l'écriture des logs dans des fichiers, la redirection de messages sur une console (la 10 ou la 12 bien souvent) ou l'envoi de messages à root.

3. Configuration de syslog

Le fichier de configuration **/etc/syslog.conf** permet de définir l'origine, l'importance et la destination de chaque message, sous forme de deux champs.

L'origine définit en fait un ensemble de **systèmes** et de **sous-systèmes** (noyau, services). La liste, extensible, est composée à l'origine des éléments suivants. L'étoile définit l'ensemble des sous-systèmes.

Sous-système	Signification
auth/authpriv	Service de sécurité et d'authentification.
cron	Service cron.
daemon	Les démons du système.
kern	Le noyau.
lpr	Le service d'impression.
mail	La messagerie.
news	Le réseau.
syslog	Syslog lui-même.
user	Messages des processus utilisateurs.
uucp	Unix to Unix CoPy.
local0->7	Messages issus de klogd, le chiffre représente le niveau.

L'importance ou **niveau** définit le niveau de sévérité du message. L'étoile définit l'ensemble de tous les niveaux. Il y a équivalence entre les niveaux émis par klogd et syslogd.

Niveau	Signification
emerg	Le système est inutilisable.
alert	Une intervention immédiate est indispensable.
crit	Erreur critique pour le sous-système.
err	Erreur de fonctionnement.
warning	Avertissement.
notice	Évènement normal méritant d'être signalé.
info	Pour information seulement.
debug	Message envoyé pour la mise au point.
none	Ignorer les messages.

La destination ou **action** peut être un fichier, un message à un utilisateur, la console, une liste d'utilisateurs... L'étoile indique tout le monde.

Les messages syslog sont inscrits dans les fichiers **/var/log/messages** et **/var/log/syslog** ou dans tout autre fichier paramétré dans **/etc/syslog.conf**.

L'exemple suivant provient d'une installation Red Hat AS 4u6.

Tout (sauf mail.*) est place dans /var/log/messages

*.info;mail.none;authpriv.none;cron.none   /var/log/messages

The authpriv file has restricted access.

authpriv.* /var/log/secure

Mails

mail.* -/var/log/maillog

Crontab

cron.* /var/log/cron

Messages d'alerte

*.emerg *

erreurs uucp et news

uucp,news.crit /var/log/spooler

Messages de boot

local7.* /var/log/boot.log

Vous pouvez vous-même, directement ou dans vos scripts, envoyer des messages à **syslogd** par la commande **logger**.

4. Les fichiers de traces

Les logs systèmes sont situés par convention dans /var/log. Tous les logs de ce répertoire ne proviennent pas de syslogd. C'est le cas par exemple des informations de connexion. Voici un exemple du contenu de ce répertoire. Il contient plusieurs fichiers textes et des répertoires. Des services peuvent décider, sans passer par **syslogd**, de concentrer et d'écrire leurs messages dans cette arborescence.

cd /var/log ; ls -l

```
-rw-r-----  1 root  root      2460 fev  7 05:34 acpid
drwxr-x---  2 root  root      4096 mar  5  2007 audit
-rw-------  1 root  root       116 mar 27 04:02 boot.log
-rw-------  1 root  root     75487 mar 28 11:10 cron
drwxr-xr-x  2 lp    sys       4096 mar 27 04:02 cups
-rw-r--r--  1 root  root     28359 fev  7 05:34 dmesg
drwx------  2 root  root      4096 aou  7  2007 httpd
-r--------  1 root  root  18747276 mar 28 11:08 lastlog
drwxr-xr-x  2 root  root      4096 jui  1  2007 mail
-rw-------  1 root  root      4537 mar 28 04:02 maillog
-rw-------  1 root  root    178348 mar 28 11:10 messages
drwx------  2 root  root      4096 oct 16 23:21 samba
-rw-------  1 root  root    214999 mar 28 11:08 secure
-rw-r--r--  1 root  root      2734 mar 28 11:01 snmpd.log
-rw-------  1 root  root         0 mar 23 04:02 spooler
drwxr-x---  2 squid squid     4096 jan 22  2007 squid
-rw-------  1 root  root     62165 mar 28 09:13 sudo.log
drwxr-xr-x  2 root  root      4096 oct  5  2004 vbox
-rw-rw-r--  1 root  utmp    127872 mar 28 11:10 wtmp
-rw-------  1 root  root     40557 mar 28 11:03 xferlog
```

TP : Les tâches administratives - Les traces du système

111.4 Automatiser les tâches d'administration (4)

Automatisation

1. Avec cron

a. Présentation

Le service **cron** permet la programmation d'événements à répétition. Il fonctionne à l'aide d'une table, appelée une **crontab**. C'est un fichier texte, éditable avec un simple éditeur, par exemple vi. Pour modifier votre crontab personnelle utilisez la commande **crontab** pour éditer la table, avec le paramètre **-e**.

Les fichiers crontabs sont sauvés dans /var/spool/cron.

Le service **cron** doit tourner pour que les crontabs soient actives.

$ ps -ef|grep cron

root 3634 1 0 18:28 ? 00:00:00 /usr/sbin/cron

b. Formalisme

Le format d'un enregistrement de crontab est le suivant :

Minutes	Heures	Jour du mois	Mois	Jour semaine	Commande
1	2	3	4	5	6

Utilisez le format suivant pour les valeurs périodiques :

Une valeur pour indiquer quand il faut exécuter la commande. Ex : la valeur 15 dans le champ minute signifie la quinzième minute.

Une liste de valeurs séparées par des virgules. Ex : 1,4,7,10 dans le champ mois pour janvier, avril, juillet, octobre.

Un intervalle de valeurs. Ex : 1-5 dans le champ jour de la semaine indique du lundi (1) au vendredi (5). Le 0 est le dimanche et le 6 le samedi.

Le caractère * pour toutes les valeurs possibles. Ex : * dans le champ jour du mois indique tous les jours du ou des mois.

c. Exemples

Exécution de df tous les jours, toute l'année, tous les quarts d'heure :

0,15,30,45 * * * * df > /tmp/libre

Exécution d'une commande tous les jours ouvrables à 17 heures :

0 17 * * 1-5 fin_travail.sh

Lister les crontabs actives :

$ crontab -l

Supprimer la crontab active :

$ crontab -r

Éditer la crontab d'un utilisateur particulier :

crontab -u user

d. crontab système

La configuration crontab générale pour le système est dans /etc/crontab.

SHELL=/bin/bash

PATH=/sbin:/bin:/usr/sbin:/usr/bin

MAILTO=root

HOME=/

run-parts

01 * * * * root run-parts /etc/cron.hourly

02 4 * * * root run-parts /etc/cron.daily

22 4 * * 0 root run-parts /etc/cron.weekly

42 4 1 * * root run-parts /etc/cron.monthly

Ici tous les jours à 4h02 du matin **run-parts /etc/cron.daily** est exécuté. Le script **run-parts** accepte en paramètre un répertoire et exécute tous les programmes présents dans ce répertoire.

$ ls cron.daily/

00-logwatch 0anacron   makewhatis.cron slocate.cron

00webalizer logrotate rpm   tmpwatch

Parmi les programmes exécutés, remarquez **logrotate** qui permet d'effectuer des sauvegardes et de renommer des fichiers logs et des journaux du système afin que ceux-ci ne deviennent pas inexploitables à cause de leur taille. Le programme **tmpwatch** est chargé de nettoyer le système des fichiers inutilisés (dans /tmp par exemple).

Enfin, le répertoire /etc/cron.d contient des crontabs supplémentaires.

e. Contrôle d'accès

Vous pouvez contrôler l'accès à la commande **crontab** par utilisateur avec les fichiers **/etc/cron.allow** et **/etc/cron.deny**.

Si cron.allow est présent, seuls les utilisateurs qui y sont explicitement indiqués peuvent utiliser **at** (cf. Automatisation - Avec at dans ce chapitre).

Si cron.allow est absent, cron vérifie la présence d'un fichier cron.deny. Tous les utilisateurs n'y étant pas sont autorisés à utiliser cron. S'il est vide la commande **cron** est autorisée pour tout le monde.

Si les deux fichiers sont absents, seul root peut utiliser cron.

f. Crontab en mode graphique

Quelques outils permettent d'éditer une crontab de manière visuelle sans passer par un éditeur de texte. L'outil kcron sous KDE est très populaire et très bien adapté pour cela.

L’outil kcron sous KDE

L’exemple de la capture présente un enregistrement de la crontab d’un utilisateur en cours d’édition. La tâche lance_backup.sh est exécutée tous les jours à 23h30. Les lignes correspondantes dans la crontab (au format texte) sont les suivantes :

Lancement de la sauvegarde

30 23 * * *   /opt/backup/lance_backup.sh

2. Avec at

a. Présentation

La commande **at** et les commandes associées permettent une gestion des traitements batchs. Contrairement à la crontab les modifications sont volatiles : elles sont perdues lorsque la session est terminée. C'est à vous de placer la liste des commandes dans un éventuel fichier et de le charger au besoin via les scripts de votre profil.

Pour que at fonctionne le service **atd** (**at daemon**) doit fonctionner.

$ ps -ef | grep atd

at 7988 1 0 21:05 ? 00:00:00 /usr/sbin/atd

b. Formalisme

Pour simplifier, il y a deux moyens d'utiliser at :

en lui passant de manière interactive une ligne de commande,

en lui passant un fichier exécutable contenant les commandes à exécuter.

Dans les deux cas, vous devez fournir à at une heure d'exécution. Le formalisme de cette heure est assez souple. Pour programmer l'exécution d'une ligne de commande à 21h20 de manière interactive :

$ at 21:20

warning: commands will be executed using /bin/sh

at> echo salut

at> <EOT>

job 4 at 2008-05-08 21:20

Après avoir saisi la ou les commandes à exécuter à 21h20, appuyez sur [Entrée] et sur une ligne vide appuyez sur [Ctrl] **D** (fin de saisie). La commande **at** confirme la programmation de la commande.

Pour programmer l'exécution d'une commande (script ou binaire) à 21h25 :

$ at -f /home/seb/test.sh 21:25

warning: commands will be executed using /bin/sh

job 6 at 2008-05-08 21:25

Heure

L'heure peut être formatée ainsi :

HHMM ou HH:MM.

L’heure peut être au format 12 ou 24h. Au format 12 heures, vous pouvez préciser AM (matin) ou PM (après-midi).

Midnight (minuit), noon (midi), teatime (16h00, typiquement anglais).

MMJJAA, MM/JJ/AA ou JJ.MM.AA pour une date précise.

Now : maintenant.

+ n minutes/hours/days/weeks : l’heure courante auquel on ajoute n minutes/heures/jours/semaines.

Si l’heure précisée est inférieure à l’heure actuelle, la commande est exécutée le lendemain.

$ at 21:30 09.05.2008

warning: commands will be executed using /bin/sh

at> echo salut !

at> <EOT>

job 9 at 2008-05-09 21:30

$ at now + 2 days

warning: commands will be executed using /bin/sh

at> echo dans deux jours

at> <EOT>

job 10 at 2008-05-10 21:29



Il existe aussi la commande **batch** qui ne prend pas d’heure. Elle exécute la commande dès que la charge de la machine l’autorise. L’heure peut être précisée, dans ce cas elle sera considérée comme « à partir de cet heure, dès que possible ».

c. Contrôle des tâches

La commande **atq** (**at queue**) permet de lister les tâches programmées :

$ atq

10 2008-05-10 21:29 a seb

9 2008-05-09 21:30 a seb

Les jobs (tâches) sont placées dans le répertoire /var/spool/atjobs, à raison de un exécutable par tâche.

```
# ls -l /var/spool/atjobs/

-rwx------ 1 seb users 5620 mai  8 21:29 a000090133cf92

-rwx------ 1 seb users 5628 mai  8 21:30 a0000a0133d531
```

Si vous regardez le contenu de l'exécutable, vous voyez que votre commande n'est pas seule. Elle est située à la fin, mais le script positionne tout l'environnement lors de la création de l'entrée at.

```
#cat a0000a0133d531

#!/bin/sh

# atrun uid=1000 gid=100

# mail      seb 0

umask 22

LESSKEY=/etc/lesskey.bin; export LESSKEY

NNTPSERVER=news; export NNTPSERVER

INFODIR=/usr/local/info:/usr/share/info:/usr/info; export INFODIR

MANPATH=/usr/local/man:/usr/share/man:/opt/gnome/share/man; export

MANPATH

KDE_MULTIHEAD=false; export KDE_MULTIHEAD

… (environ 80 lignes) …

cd /home/seb || {

        echo 'Execution directory inaccessible' >&2

        exit 1

}

echo salut !
```

La commande **atrm** permet de supprimer une tâche :

$ atrm 10

$ atrm 9

$ atq

d. Contrôle d'accès

Vous pouvez contrôler l'accès à la commande **at** par utilisateur avec les fichiers **/etc/at.allow** et **/etc/at.deny**.

Si at.allow est présent, seuls les utilisateurs qui y sont explicitement indiqués peuvent utiliser at.

Si at.allow est absent, at vérifie la présence d'un fichier at.deny. Tous les utilisateurs n'y étant pas sont autorisés à utiliser at. S'il est vide la commande at est autorisée pour tout le monde.

Si les deux fichiers sont absents, seul root peut utiliser at.

TP : Les tâches administratives - Automatisation des tâches

111.5 Les sauvegardes (5)

Archivage et backup

1. Les outils de sauvegarde

La sauvegarde est un travail important de l'administrateur puisqu'en cas de gros problème, on passe généralement par une restauration du système depuis une sauvegarde, ou une image du système lorsque celui-ci était encore intègre (bon fonctionnement, pas de corruption). Chaque Unix est fourni avec des commandes et des procédures de sauvegarde qui lui sont propres. On distingue tout de même quelques outils communs.

a. Commandes, plans, scripts

Pour la sauvegarde de fichiers et d'arborescences, utilisez les commandes **tar** et **cpio**. Ces commandes sauvent une arborescence, et pas un système de fichiers. On peut faire coïncider les deux.

Pour la sauvegarde physique de disques et de systèmes de fichiers (des dumps), utilisez la commande **dd**.

Une sauvegarde incrémentale consiste à sauvegarder une première fois la totalité des données, puis ensuite uniquement les fichiers modifiés. On trouve aussi sous forme de logiciels libres ou dans le commerce des solutions plus pointues de sauvegarde (Networker par exemple).

L'administrateur aura parfois à définir des scripts de sauvegarde et de restauration adaptés au cas par cas (partition système, données applicatives...) et à automatiser quand c'est possible l'exécution de ceux-ci en fonction de la date, l'heure ou la charge de la machine.

Il sera aussi très important de définir un plan de sauvegarde, en se posant les bonnes questions :

Que faut-il sauvegarder ?

Avec quelle fréquence ?

Combien de temps conservera-t-on les sauvegardes, à quel endroit, en combien d'exemplaires ?

À quel endroit sera stocké l'historique des sauvegardes ?

Quel est le support le plus approprié ?

Quels sont les besoins, en capacité, du support de sauvegarde ?

Combien de temps prévoit-on pour sauvegarder un fichier, un système de fichiers et est-ce raisonnable ?

La sauvegarde doit-elle être automatique ou manuelle ?

Quelle est la méthode de sauvegarde la plus appropriée ?

Chaque cas étant unique, cet ouvrage ne peut répondre à toutes ces questions. Les réponses dépendent de l'environnement cible (production, intégration, tests, etc.). Cependant envisagez toujours la mise en place d'une sauvegarde système (racine, /opt, /usr, /var, /boot, etc.) après une installation et avant une modification importante, au cas où il faudrait revenir en arrière.

b. Voici quelques autres commandes

mt : contrôle d'une bande magnétique.

touch : met la date de dernière modification à l'heure actuelle, pour forcer une sauvegarde incrémentale.

find : sélectionne les fichiers à sauvegarder.

compress et **uncompress** : compression et décompression des fichiers.

gzip, **gunzip**, **zcat**, compression et décompression au format GnuZip.

2. Tar

La commande **tar** est simple et efficace. Elle crée des archives des fichiers, y compris l'arborescence de fichiers, sur tout type de support y compris dans une autre fichier (archive à l'extension .tar). L'archive ainsi créée peut s'étendre sur plusieurs volumes : quand la bande ou la disquette est pleine, c'est à l'utilisateur d'en insérer une nouvelle et la sauvegarde/restitution continue.

a. Archiver

La syntaxe est la suivante :

tar cvf nom_archive Fichier(s)

Par exemple pour placer dans une archive tar le répertoire Desktop :

$ tar cvf desktop.tar Desktop/

Desktop/

Desktop/fusion-icon.desktop

Desktop/konsole.desktop

Desktop/Support.desktop

Desktop/Office.desktop

Desktop/Terminal.desktop

Desktop/MozillaFirefox.desktop

Desktop/Printer.desktop

Desktop/.directory

Desktop/myComputer.desktop

Desktop/trash.desktop

Desktop/SuSE.desktop

Desktop/Windows.desktop

Les paramètres sont les suivants :

c : création d' archive,

v : mode bavard, tar indique ce qu'il fait,

f : le paramètre suivant est le nom de l'archive.

b. Lister

La syntaxe est :

tar tvf nom_archive

Pour lister le contenu de l'archive précédente :

```
$ tar tvf desktop.tar
drwx------ seb/users      0 2008-04-17 09:44 Desktop/
-rw-r--r-- seb/users    191 2007-10-20 20:10 Desktop/fusion-icon.desktop
-rw-r--r-- seb/users   4786 2007-09-26 00:43 Desktop/konsole.desktop
-rw-r--r-- seb/users    665 2008-04-08 15:14 Desktop/Support.desktop
-rw-r--r-- seb/users   1051 2007-10-05 10:16 Desktop/Office.desktop
-rw-r--r-- seb/users   4586 2007-12-05 11:37 Desktop/Terminal.desktop
-rw-r--r-- seb/users    829 2007-10-17 12:12 Desktop/MozillaFirefox.desktop
-rw-r--r-- seb/users   3952 2007-10-05 10:16 Desktop/Printer.desktop
-rw-r--r-- seb/users   2053 2007-10-05 10:16 Desktop/.directory
-rw-r--r-- seb/users    450 2007-10-23 11:58 Desktop/myComputer.desktop
-rw-r--r-- seb/users    218 2008-02-22 08:43 Desktop/trash.desktop
-rw-r--r-- seb/users    328 2008-04-08 15:14 Desktop/SuSE.desktop
-rw-r--r-- seb/users    472 2008-04-17 09:44 Desktop/Windows.desktop
```

Le paramètre **t** liste le contenu de l'archive.

c. Restauration

Pour restaurer le contenu d'une archive la syntaxe est :

tar xvf nom_archive fichiers

Pour restaurer l'archive précédente :

tar xvf desktop.tar

Desktop/

Desktop/fusion-icon.desktop

Desktop/konsole.desktop

Desktop/Support.desktop

Desktop/Office.desktop

Desktop/Terminal.desktop

Desktop/MozillaFirefox.desktop

Desktop/Printer.desktop

Desktop/.directory

Desktop/myComputer.desktop

Desktop/trash.desktop

Desktop/SuSE.desktop

Desktop/Windows.desktop

Le paramètre **x** permet l'extraction de l'ensemble des fichiers de l'archive, ou du ou des fichiers spécifiés à la suite du nom de l'archive.

d. Autres paramètres

La commande **tar** de gnu permet de gérer les formats de compression directement :

z : l'archive est compressée au format gzip.

Z : l'archive est compressée au format compress.

j : l'archive est compressée au format bzip2.

Ainsi les commandes précédentes pour le format de compression gzip deviennent :

$ tar cvzf desktop.tar.gz Desktop/

Desktop/

Desktop/fusion-icon.desktop

Desktop/konsole.desktop

Desktop/Support.desktop

Desktop/Office.desktop

Desktop/Terminal.desktop

Desktop/MozillaFirefox.desktop

Desktop/Printer.desktop

Desktop/.directory

Desktop/myComputer.desktop

Desktop/trash.desktop

Desktop/SuSE.desktop

Desktop/Windows.desktop

$ ls -l desktop.tar*

-rw-r--r-- 1 seb users 30720 mai 9 11:16 desktop.tar

-rw-r--r-- 1 seb users 7556 mai 9 11:22 desktop.tar.gz

Notez la différence de taille. Les options de compression peuvent être utilisées avec c, t et x. Notez que c'est l'archive finale qui est compressée, par les fichiers individuellement. Il peut être préférable de ne pas spécifier d'option de compression si vous sauvez sur une bande dont le lecteur gère lui-même la compression.

Si votre archive est compressée et qu'elle est à destination d'un autre système, ou que vous souhaitez garder une compatibilité avec les paramètres par défaut de tar, vous pouvez procéder comme ceci :

$ gzip -cd desktop.tar.gz | tar xvf -

Desktop/

Desktop/fusion-icon.desktop

Desktop/konsole.desktop

Desktop/Support.desktop

Desktop/Office.desktop

Desktop/Terminal.desktop

Desktop/MozillaFirefox.desktop

Desktop/Printer.desktop

Desktop/.directory

Desktop/myComputer.desktop

Desktop/trash.desktop

Desktop/SuSE.desktop

Desktop/Windows.desktop

Le paramètre **-d** précise à **gzip** de décompresser le fichier, tandis que **-c** passe le résultat par la sortie standard. Le **-** final indique à tar de récupérer le flux par l'entrée standard.

3. cpio

La commande **cpio** sauvegarde sur la sortie standard les fichiers dont on saisit les noms sur l'entrée standard, par défaut l'écran et le clavier. Vous devez donc utiliser les redirections. Cpio ne compresse pas les archives. C'est à vous de le faire.

a. Archiver

La syntaxe générale est :

cpio -oL

Les paramètres les plus utilisés sont :

-o : output, création de la sauvegarde en sortie.

-L : sauve les fichiers liés et pas les liens symboliques.

-v : mode bavard « verbose », informations détaillées.

-c : sauvegarde des attributs des fichiers sous forme ASCII (pour l'échange entre divers OS).

Voici comment archiver et compresser le répertoire Desktop :

find Desktop -print| cpio -ocv | gzip > archive.cpio.gz

Desktop

Desktop/fusion-icon.desktop

Desktop/konsole.desktop

Desktop/Support.desktop

Desktop/Office.desktop

Desktop/Terminal.desktop

Desktop/MozillaFirefox.desktop

Desktop/Printer.desktop

Desktop/.directory

Desktop/myComputer.desktop

Desktop/trash.desktop

Desktop/SuSE.desktop

Desktop/Windows.desktop

42 blocks

> ls -l archive.cpio.gz

-rw-r--r-- 1 seb users 7377 mai 9 11:33 archive.cpio.gz

b. Lister

La syntaxe générale est :

cpio -it archive

Les paramètres sont :

-i : lecture de l'archive en entrée.

-t : comme pour tar, liste le contenu de l'archive.

$ cat archive.cpio.gz | gzip -cd | cpio -it

Desktop

Desktop/fusion-icon.desktop

Desktop/konsole.desktop

Desktop/Support.desktop

Desktop/Office.desktop

Desktop/Terminal.desktop

Desktop/MozillaFirefox.desktop

Desktop/Printer.desktop

Desktop/.directory

Desktop/myComputer.desktop

Desktop/trash.desktop

Desktop/SuSE.desktop

Desktop/Windows.desktop

42 blocks

c. Restaurer

La syntaxe générale est :

cpio -i[umd]

-u : restauration inconditionnelle, avec écrasement des fichiers qui existent déjà. Par défaut les fichiers ne sont pas restaurés si ceux présents sur le disque sont plus récents ou du même âge.

-m : les fichiers restaurés conservent leur dernière date de modification.

-d : cpio reconstruit l'arborescence des répertoires et sous-répertoires manquants.

Pour restaurer l'archive précédente :

$ cat archive.cpio.gz | gzip -cd | cpio -iuvd

Desktop

Desktop/fusion-icon.desktop

Desktop/konsole.desktop

Desktop/Support.desktop

Desktop/Office.desktop

Desktop/Terminal.desktop

Desktop/MozillaFirefox.desktop

Desktop/Printer.desktop

Desktop/.directory

Desktop/myComputer.desktop

Desktop/trash.desktop

Desktop/SuSE.desktop

Desktop/Windows.desktop

42 blocks

4. dd

La commande **dd** (**device to device**) est destinée à la copie physique, bloc par bloc, d'un fichier périphérique vers un fichier périphérique ou quelconque. À l'origine elle était utilisée pour la lecture et l'écriture sur bande magnétique, mais elle peut être employée avec n'importe quel fichier. La commande **dd** permet de réaliser des copies physiques de disques et de systèmes de fichiers.

Argument	Rôle
if=fichier	Nom du fichier en entrée (celui à copier).
of=fichier	Nom du fichier en sortie.
bs=n	Taille du bloc en octets.
count=n	Nombre de blocs à copier.
skip=n	Nombre de bloc à sauter au début du fichier d'entrée.
conv=	Conversion de l'entrée.
seek=	Nombre de blocs à sauter au début du fichier de sortie.
-s	Shell (commande de connexion) par défaut de l'utilisateur (variable SHELL). L'utilisateur peut le changer via la commande **chsh**.
-p	Le mot de passe de l'utilisateur. Attention ! le mot de passe doit déjà être crypté ! Aussi à moins de recopier le mot de passe d'un compte générique, vous préférerez utiliser ensuite la commande **passwd**.

L'option **conv** admet les paramètres suivants :

ascii : convertir l'EBCDIC en ASCII.

ebcdic : convertir l'ASCII en EBCDIC.

block : compléter les blocs se terminant par un saut de ligne avec des espaces, jusqu'à atteindre la taille mentionnée par bs.

unblock : remplacer les espaces en fin de blocs (de taille cbs) par un saut de ligne.

lcase : transformer les majuscules en minuscules.

ucase : transformer les minuscules en majuscules.

noerror : continuer même après des erreurs de lecture.

notrunc : ne pas limiter la taille du fichier de sortie.

sync : compléter chaque bloc lu avec des NULs pour atteindre la taille ibs.

Ici vous allez placer le secteur de boot de la partition (où est installé lilo ou grub) dans un fichier. Le fichier ainsi créé pourra être utilisé avec le chargeur de NT/2000/XP pour démarrer sous Linux.

dd if=/dev/sda1 of=boot.lnx bs=442 count=1

Pour créer un fichier vide d'une taille de 1Mo :

$ dd if=/dev/zero of=vide bs=1024 count=1024

1024+0 enregistrements lus

1024+0 enregistrements écrits

1048576 bytes (1,0 MB) copied, 0,0199192 s, 52,6 MB/s

TP : Les tâches administratives – Archivage

111.6 Maintenir le système à l'heure (4)

L'horloge

1. Connaître l'heure

a. date

Pour connaître l'heure, utilisez la commande **date**. Elle permet de donner la date actuelle, mais aussi de calculer d'autres dates en fonction soit de la date actuelle, soit en fonction d'une date quelconque. Date permet aussi de modifier la date et l'heure du système

$ date

sam mai 10 13:58:38 CEST 2008

Par défaut la date affichée est la date (et l'heure) locale, configurée en fonction du fuseau horaire. Pour afficher l'heure UTC :

$ date --utc

sam mai 10 11:01:10 UTC 2008

Le format de la date peut être modifié à volonté à l'identique de ce qui peut se faire avec la fonction C strftime. Dans ce cas la syntaxe est :

date +"format".

Voici quelques exemples de format possible :

Format	Résultat
%H	L'heure au format 00..23.
%M	Minutes 00..59.
%S	Secondes 00..60.
%T	Heure actuelle sur 24 heures.
%r	Heure actuelle sur 12 heures.
%Z	Fuseau horaire.
%a	Jour abrégé (lun, mar, etc.).
%A	Jour complet.
%b	Mois abrégé.
%B	Mois complet.
%d	Jour du mois.
%j	Jour de l'année.
%m	Numéro du mois.
%U	Numéro de la semaine 00..53.
%y	Deux derniers chiffres de l'année.
%Y	Année complète.

Pour afficher une date complète :

$ date +"Nous sommes le %A %d %B %Y, il est %H heures, %M minutes et

%S secondes"

Nous sommes le samedi 10 mai 2008, il est 14 heures, 10 minutes et

20 secondes

Vous pouvez modifier la base de calcul en passant le paramètre **--date** suivi d'une date ou d'un calcul. Les mots clés today, yesterday, tomorrow, day(s), week(s), month(es), year(s),

hour(s), minute(s), second(s) sont acceptés, avec + (ajout à la date) ou - ou ago (retranche à la date précisée). Si la date n'est pas précisée, c'est la date en cours.

Dans 10 jours :

date --date "10 days"

mar mai 20 13:13:05 CEST 2008

Demain :

date --date "tomorrow"

Hier :

date --date "yesterday"

Une semaine après Noël 2008 :

date --date "12/25/2008 23:59:00 + 1 week"

jeu jan 1 23:59:00 CET 2009

b. hwclock

La commande **hwclock** permet d'interroger directement l'horloge matérielle RTC. Le paramètre **--show** (par défaut) affiche la date actuelle. Elle est différente du temps système provenant de ntp ou date. La fin de l'affichage donne d'ailleurs le décalage.

hwclock --show

sam 10 mai 2008 13:16:14 CEST -0.390436 secondes

Il n'est pas possible de formater le résultat de la commande.

2. Modifier l'horloge matérielle

L'horloge matérielle peut être modifiée uniquement en tant que root via les commandes **date** (horloge système interne au noyau) et **hwclock** (horloge matérielle).

a. Via date

Modifiez la date et l'heure en passant le paramètre **-s** :

date -s "05/09/2008 14:00"

ven mai 9 14:00:00 CEST 2008

date

ven mai 9 14:00:03 CEST 2008

b. Via hwclock

La commande **hwclock** modifie l'horloge matérielle (RTC) et/ou l'horloge système. L'heure matérielle étant indépendante de l'heure système, les résultats peuvent surprendre :

hwclock --set --date "05/10/2008 14:00"

date

sam mai 10 13:30:53 CEST 2008

hwclock

sam 10 mai 2008 14:00:13 CEST -0.658230 secondes

Vous pouvez synchroniser l'heure système et l'heure matérielle dans les deux sens. Pour que l'heure matérielle soit synchronisée à partir de l'heure système :

hwclock --systohc

hwclock

sam 10 mai 2008 13:34:00 CEST -0.931220 secondes

Pour effectuer l'inverse :

hwclock --hctosys

3. NTP

a. Principe

NTP (**Network Time Protocol**) est un protocole qui permet de synchroniser les horloges des ordinateurs via le réseau, notamment TCP/IP et donc Internet. Nos ordinateurs utilisant des horloges au quartz, celles-ci, selon la qualité des composants, peuvent parfois fortement et rapidement avancer ou retarder.

Il y a de nombreux domaines où il est inadmissible de ne pas avoir un système à l'heure notamment pour des raisons de synchronisation très précises.

Un serveur NTP diffuse l'heure au format UTC. Le client récupère l'heure et l'adapte en fonction de son fuseau horaire. Le serveur ne gère pas non plus les changements d'heure.

Pour peu que le serveur NTP soit à jour, l'heure est très précise. Elle est codée sur 64 bits :

les 32 premiers bits donnent le nombre de secondes depuis le 1er janvier 1900 à minuit (donc le bug NTP aura lieu avant le bug Unix) ;

les 32 derniers bits donnent la précision des secondes.

Le nouveau protocole NTP4 donne une précision des secondes sur 64 bits, évitant ainsi un bug gênant dans le futur.

Vous trouverez à l'URL suivante une liste de serveurs NTP français.

http://www.cru.fr/services/ntp/serveurs_francais0

b. Client ntp

Le service ntpd permet de synchroniser une machine auprès d'un serveur de temps.

$ ps -ef|grep ntp

root   6523 5378 0 14:04 pts/2   00:00:00 ntpd

Le fichier de configuration est **/etc/ntpd.conf**. En principe ce fichier contient déjà un certain nombre de lignes qu'il faut éviter de toucher. Vous pouvez, voire devez, rajouter une ligne pointant sur le serveur de temps que vous avez choisi (par exemple chronos.espci.fr) :

server chronos.espci.fr

Relancez le service. Votre machine doit se mettre à l'heure.

Vous pouvez forcer une synchronisation manuelle avec la commande **ntpdate**. Celle-ci prend pour paramètre un nom de serveur ntp.

ntpdate chronos.espci.fr 

10 May 14:09:21 ntpdate[6551]: adjust time server 193.54.82.20 off-

set 0.154057 sec

Si vous ne souhaitez pas utiliser le service ntpd, vous pouvez placer cette commande en crontab tous les jours, ou toutes les heures.

THEME : 112 Les bases du réseau

112.1 Les bases de TCP/IP (4)

TCP/IP

1. Bases

L’origine de **TCP/IP** provient des recherches du **DARPA (Defense Advanced Research Project Agency)** qui débutent en 1970 et débouchent sur **ARPANET**. Dans les faits, le DARPA a financé l’université de Berkeley qui a intégré les protocoles de base de TCP/IP au sein de son système **UNIX BSD 4**.

TCP/IP s’est popularisé grâce à son interface générique de programmation d’échanges de données entre les machines d’un réseau, les primitives **sockets**, et l’intégration de protocoles applicatifs. Les protocoles de TCP/IP sont supervisés par l’**IAB (Internet Activities Board)** lui-même supervisant deux autres organismes :

L’**IRTF** (**Internet Reseach Task Force**) qui est responsable du développement des protocoles.

L’**IETF** (**Internet Engineering Task Force**) qui est responsable du réseau Internet.

Les adresses réseau sont distribuées par le **NIC** (**Network Information Center**) et en France l’**INRIA**. L’ensemble des protocoles de TCP/IP est décrit dans les documents **RFC** (**Request For Comments**) (voir le RFC 793).

La couche inférieure est **IP** (**Internet Protocol**).

La couche de transport est **TCP** (**Transmission Control Protocol**) ou **UDP** (**User Datagram Protocol**).

Les couches supérieures sont les couches des protocoles applicatifs, par exemple :

NFS (**Network File System**) : partage de fichiers à distance.

DNS (**Domain Name System**) : association hôte<->IP.

FTP (**File Transfer Protocol**) : transfert de fichiers.

TELNET : émulation d’un terminal de type texte...

La version du protocole IP représenté est la V4. Le futur, déjà présent, est le protocole IPV6. Compatible IPV4, il propose un adressage sur 128 bits (16 octets) permettant d’étendre les capacités du réseau notamment en matière de taille et d’adressage.

2. Adressage

a. Classes

Il est important de savoir avant l'installation dans quel type de réseau doit s'intégrer le nouveau serveur, TCP/IP bien sûr, mais il faut déjà lui réserver une adresse IP, un hostname (nom de machine réseau), connaître les diverses passerelles, le nom de domaine, la classe utilisée et le masque de sous-réseau netmask.

Voici un bref rappel sur les classes IP. Une adresse IP est définie sur 32 bits et représentée par quatre nombres séparés par des points : **n1.n2.n3.n4**. Cette adresse est constituée de deux parties qui définissent l'adresse réseau et l'hôte dans le réseau.

Distinguez, suivant les cas, quatre ou cinq classes d'adresses : A, B, C, D et E, mais seules les trois premières nous intéressent.

Légende : N et h sont des bits, N identifiant du réseau h identifiant de la machine.

Classe A : 0NNNNNNN hhhhhhhh hhhhhhhh hhhhhhhh soit 1.x.x.x à

126.x.x.x.

n1 est compris entre 1 et 126.

16777214 hôtes, 127 réseaux.

Classe B : 10NNNNNN NNNNNNNN hhhhhhhh hhhhhhhh soit de 128.0.x.x à

191.255.x.x.

n1 est compris entre 128 et 191.

65534 hôtes, 16382 réseaux.

Classe C : 110NNNNN NNNNNNNN NNNNNNNN hhhhhhhh soit de 192.0.0.x à

223.255.255.x.

n1 est compris entre 192 et 223.

254 hôtes, 2097150 réseaux.

Classe D : Commence par 1110, pour la multidiffusion IP**.**

Classe E : Commence par 1111, pour expérimentation.

Il existe des adresses d'hôtes qui ne peuvent pas être exploitées. Par exemple dans la classe C on ne peut avoir que 254 hôtes, alors que l'identifiant de la machine est codé sur 8 bits (donc 256 valeurs). C'est que l'adresse 0 représente l'adresse du réseau, et l'adresse 255 celle du **broadcast** (multidiffusion).

Notez que les adresses suivantes ne doivent pas être routées sur Internet et sont réservées aux réseaux locaux.

10.0.0.0 - 10.255.255.255 (10/8)

172.16.0.0 - 172.31.255.255 (172.16/12)

192.168.0.0 - 192.168.255.255 (192.168/16)

L'adresse 127.0.0.1 est l'adresse de loopback ou bouclage : elle représente la machine elle-même, ainsi que le sous-réseau127.0.0.0/8.

b. Sous-réseaux

De plus, il est possible de découper ces réseaux en sous-réseaux à l'aide de masques permettant un découpage plus fin des adresses. Un **netmask** est un masque binaire qui permet de séparer immédiatement l'adresse du réseau et du sous-réseau de l'adresse de l'hôte dans l'adresse IP globale. Les masques prédéfinis sont :

Classe A : 255.0.0.0

Classe B : 255.255.0.0

Classe C : 255.255.255.0

Pour communiquer directement entre eux les hôtes doivent appartenir à un même réseau ou sous-réseau. Calculer un sous-réseau est assez simple. Voici un exemple pour un réseau de classe C.

Réseau : 192.168.1.0

Adresse de réseau : 192.168.1.255

Masque de réseau : 255.255.255.0

Calculer un masque de sous-réseau :

Pour calculer le masque de sous-réseau, vous devez tout d'abord déterminer combien de machines vous souhaitez intégrer dans celui-ci. Un réseau de classe C permet d'intégrer 254 machines (0 et 255 étant réservés). Vous souhaitez créer des réseaux contenant 60 machines.

Ajoutez 2 à cette valeur pour les adresses réservées (adresse du sous-réseau et adresse de broadcast) ce qui donne **62**.

Une fois le nombre de machines déterminé, trouvez la puissance de deux exacte ou juste supérieure au nombre trouvé. 2 puissance 6 donne **64**.

Écrivez le masque en binaire, placez tous les bits du masque de réseau de classe C à 1 et placez à 0 les 6 premiers bits du masque correspondant à la partie machine : **11111111 11111111 11111111 11000000**

Convertissez ce masque en décimal : **255.255.255.192**, et calculez l'ensemble des sous-réseaux possibles. Comme vous êtes dans un réseau de classe C, vous pouvez encore faire varier les deux derniers bits de la partie machine :

00xxxxxx : 255.255.255.0

01xxxxxx : 255.255.255.64

10xxxxxx : 255.255.255.128

11xxxxxx : 255.255.255.192

Au final, vous obtenez quatre sous-réseaux de 62 machines, soit 248 machines. Vous tombez bien sur 256 si vous rajoutez les quatre adresses de broadcast et les quatre adresses de réseau.

c. Routage

Le masque de réseau permet de déterminer si une machine destinataire est sur le même réseau que vous ou non. Il faut indiquer le chemin que doivent prendre les paquets IP pour rejoindre leur destination. Si votre machine est un poste client disposant d'une seule carte réseau et que ce réseau ne comporte qu'un seul routeur (cas classique d'une connexion vers Internet) alors vous devez créer deux routes. La première est celle indiquant quelle carte réseau doivent emprunter les paquets pour accéder au reste du réseau (au sous-réseau), la seconde quelle route doivent emprunter les paquets pour sortir du réseau. Généralement, on parle de route par défaut quand un seul routeur est présent.

Vers réseau1 -> utiliser interface réseau gauche.

Vers réseau2 -> utiliser interface réseau droite.

Vers autres -> utiliser interface réseau droite vers routeur1.

Exemple : Réseau1 de classe C 192.168.1.0 sur eth0, Réseau2 de classe B 172.16.0.0 sur eth1, adresse de routeur 192.168.1.254.

Réseau	Masque	Interface	Passerelle
192.168.1.0	255.255.255.0	eth0	eth0

172.16.0.0	255.255.0.0	eth1	eth1
0.0.0.0	0.0.0.0	eth0	192.168.1.254

Tous les paquets réseaux vers 192.168.1.0 transiteront par eth0. Tous les paquets à destination de 172.16.0.0 transiteront par eth1. Par défaut, tous les autres paquets pour les réseaux non spécifiés transiteront par eth0 et seront traités par la passerelle 192.168.1.254 qui routera les paquets.

TP : Le réseau - Configuration TCP/IP de Linux

112.3 Configuration TCP/IP de Linux (7)

Outils réseaux

a. FTP

Il est utile de connaître la commande **ftp** (**file transfer protocol**). Elle permet le transfert de fichiers entre deux machines. Elle prend comme paramètre le nom de la machine distante. Pour que la commande **ftp** fonctionne, il faut que le service ftp fonctionne sur la machine distante et sur le port 21.

Voici un exemple (peu pratique) de connexion avec erreur et nouvel essai.

ftp> open

(to) machine

Connected to machine.

220 machine FTP server (Digital UNIX Version 5.60) ready.

Name (machine:root): root

331 Password required for root.

Password:

530 Login incorrect.

Login failed.

Remote system type is UNIX.

Using binary mode to transfer files.

ftp> user

(username) root

331 Password required for root.

Password:

230 User root logged in.

ftp> pwd

257 "/" is current directory.

Le plus simple est tout de même :

$ ftp machine

Connected to machine.

220 machine FTP server (Digital UNIX Version 5.60) ready.

Name (machine:root): root

331 Password required for root.

Password:

230 User root logged in.

Remote system type is UNIX.

Using binary mode to transfer files.

ftp>

Voici une liste de commandes ftp.

Commande	Action
open	Suivi d'un nom de machine, ouvre une connexion sur la machine spécifiée.
user	Saisie de l'utilisateur distant pour une connexion.
quit	Fin de la connexion et fin de la commande ftp.
ascii	Transfert des fichiers en mode ASCII (conversion des caractères spéciaux et fin de ligne en MS et Unix par exemple).
binary	Transfert des fichiers en mode binaire.
glob	Supprime l'interprétation des caractères spéciaux.
help	Affiche l'aide.
prompt	Suivi de on ou off, active ou désactive la confirmation individuelle de transfert pour chaque fichier (mget ou mput).

Commande	Action
pwd	Affiche le répertoire distant courant.
cd	Suivi du chemin, déplacement dans l'arborescence distante.
ls	Liste les fichiers de la machine distante.
delete	Suivi d'un nom de fichier, supprime le fichier distant.
mdelete	Multiple. Supprime les fichiers distants.
get	Récupère le fichier distants.
mget	Multiple. Récupère les fichiers distants (liste ou modèle).
put	Envoie le fichier local vers la machine distante.
mput	Multiple. Envoie les fichiers locaux sur la machine distante (liste ou modèle).
close/disconnect	Ferme la session actuelle.
lcd	Change de répertoire sur la machine locale.
hash	Durant les transferts, écrit un « # » sur écran pour chaque buffer transféré.
system	Informations sur le système distant.
recv	Réception d'un fichier.
send	Envoi d'un fichier.
rename	Renomme un fichier distant.
mkdir	Crée un répertoire sur la machine distante.
rmdir	Supprime un répertoire sur la machine distante.
!commande	Exécute la commande locale.

b. Telnet

Telnet est un client léger permettant d'ouvrir une connexion et une session sur une machine distante proposant un serveur telnet. Ce serveur est souvent lancé depuis xinetd ou inetd. Sa syntaxe est très simple :

$ telnet -l user machine port

Exemple :

telnet 192.168.1.60

Trying 192.168.1.60...

Connected to 192.168.1.60.

Escape character is '^]'.

Welcome to openSUSE 10.3 (i586) - Kernel 2.6.24.4-default (3).

slyserver login: seb

Mot de passe :

Dernière connexion : jeudi 15 mai 2008 à 06:26:30 CEST de console

sur :0

Vous avez un nouveau message.

Have a lot of fun...

seb@slyserver:~>



Attention ! Le service (et le client) telnet n'est absolument pas sécurisé : les connexions transitent en clair sur le réseau, et n'importe quel renifleur IP (wireshark par exemple) peut intercepter et voir tout ce qui est fait. Même le mot de passe est transmis en clair (en texte). Évitez d'utiliser ce service et concentrez-vous sur OpenSSH.

c. Ping

La commande **ping** est une commande centrale, voire incontournable. La première chose que l'on fait généralement pour savoir si une machine est accessible ou non, c'est d'essayer de la « pinguer » (sous réserve que la configuration du firewall autorise les requêtes ICMP).

Ping émet un « écho » réseau, un peu comme un sonar, et attend une réponse, le retour de l'écho. Il utilise pour cela le protocole ICMP. Interrompez la commande ping avec [Ctrl] **C**.

$ ping www.kde.org

PING www.kde.org (62.70.27.118) 56(84) bytes of data.

64 bytes from 62.70.27.118: icmp_seq=1 ttl=57 time=10.5 ms

64 bytes from 62.70.27.118: icmp_seq=2 ttl=57 time=11.3 ms

64 bytes from 62.70.27.118: icmp_seq=3 ttl=57 time=10.4 ms

64 bytes from 62.70.27.118: icmp_seq=4 ttl=57 time=11.5 ms

...

Trois paramètres doivent attirer votre attention :

-c permet de préciser le nombre d'échos à émettre.

-b permet d'émettre un écho sur une adresse de broadcast.

-I permet de spécifier l'interface réseau.

Dans le premier cas le paramètre peut être utile pour tester dans un script si un serveur répond :

ping -c 1 10.9.238.170 >/dev/null 2>&1 && echo "Le serveur répond"

Le serveur répond

Dans le second cas, toutes les adresses du sous-réseau concerné par l'adresse de broadcast doivent répondre.

ping -b 192.168.1.255

WARNING: pinging broadcast address

PING 192.168.1.255 (192.168.1.255) 56(84) bytes of data.

64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=0.232 ms

64 bytes from 192.168.1.60: icmp_seq=1 ttl=64 time=0.240 ms

64 bytes from 192.168.1.130: icmp_seq=1 ttl=255 time=0.285 ms

64 bytes from 192.168.1.139: icmp_seq=1 ttl=255 time=0.292 ms

…

Dans le dernier cas, vous pouvez spécifier une carte de sortie. Cette option est très utile pour vérifier une résolution DNS ou une route.

ping -I eth0 192.168.1.60

PING 192.168.1.60 (192.168.1.60) from 192.168.1.10:eth0: 56(84) bytes

of data.

64 bytes from 192.168.1.60: icmp_seq=1 ttl=62 time=0.478 ms

64 bytes from 192.168.1.60: icmp_seq=2 ttl=62 time=0.408 ms

…

d. Traceroute

Quand vous tentez d'accéder à un hôte distant depuis votre machine, les paquets IP passent souvent par de nombreuses routes, parfois différentes selon le point de départ et de destination, l'engorgement, etc. Le trajet passe par de nombreuses passerelles (gateways), qui dépendent des routes par défaut ou prédéfinies de chacune d'elles.

La commande **traceroute** permet de visualiser chacun des points de passage de vos paquets IP à destination d'un hôte donné. Dans l'exemple suivant, l'hôte situé en région parisienne sur le réseau du fournisseur Free tente de déterminer la route empruntée pour se rendre sur le serveur www.kde.org. L'adresse IP source (hors réseau local) est volontairement masquée.

$ traceroute www.kde.org

traceroute to www.kde.org (62.70.27.118), 30 hops max, 40 byte packets

1 DD-WRT (192.168.1.1) 0.558 ms   0.533 ms   0.585 ms

2 82.xxx.yyy.zzz (82.xxx.yyy.zzz) 6.339 ms   6.404 ms   6.901 ms

3 * * *

4 * * *

5 212.73.205.5 (212.73.205.5) 39.267 ms   35.499 ms   31.736 ms

6 ae-12-55.car2.Paris1.Level3.net (4.68.109.144) 6.485 ms ae-22-

52.car2.Paris1.Level3.net (4.68.109.48) 6.401 ms   6.338 ms

7 UUnet-Level3.Level3.net (212.73.240.206) 6.113 ms   6.152 ms

5.866 ms

8 so-3-2-0.TL2.PAR2.ALTER.NET (146.188.8.121) 6.107 ms   6.410 ms

6.365 ms

9 so-2-2-0.TL2.STK2.ALTER.NET (146.188.7.33) 87.323 ms   86.840 ms

87.010 ms

10 so-7-1-0.XR2.OSL2.ALTER.NET (146.188.15.62) 96.491 ms   97.148 ms

96.488 ms

11 ge-0-1-0.GW6.OSL2.ALTER.NET (146.188.3.242) 95.972 ms   95.934 ms

96.108 ms

12 213.203.63.74 (213.203.63.74) 95.320 ms   94.321 ms   96.188 ms

13 leeloo.troll.no (62.70.27.10) 94.064 ms   94.052 ms   92.374 ms

14 jamaica.kde.org (62.70.27.118) 97.064 ms   96.182 ms   97.853 ms

e. Whois

Savez-vous que vous pouvez obtenir toutes les informations voulues sur un domaine (toto.fr) à l'aide de la commande **whois** ? Par exemple, pour obtenir toutes les informations sur le domaine kde.org :

> whois kde.org

…

Domain ID:D1479623-LROR

Domain Name:KDE.ORG

Created On:14-Dec-1996 05:00:00 UTC

Last Updated On:12-Oct-2007 13:10:18 UTC

Expiration Date:13-Dec-2012 05:00:00 UTC

Sponsoring Registrar:easyDNS Technologies Inc. (R1247-LROR)

Status:CLIENT TRANSFER PROHIBITED

Status:CLIENT UPDATE PROHIBITED

Registrant ID:tu2YDGaiunEvz5QA

Registrant Name:Trolltech AS

Registrant Organization:Trolltech AS

Registrant Street1:Sandakerveien 116, PO Box 4332 Nydalen

Registrant Street2:

Registrant Street3:

Registrant City:Oslo

Registrant State/Province:N/A

Registrant Postal Code:N-0402

Registrant Country:NO

Registrant Phone:+1.4721604800

Registrant Phone Ext.:

Registrant FAX:+1.4721604801

Registrant FAX Ext.:

Registrant Email:hostmaster@trolltech.com

Admin ID:tubEUVkFfutkJZMD

Admin Name:Trolltech AS

Admin Organization:Trolltech AS

Admin Street1:Sandakerveien 116, PO Box 4332 Nydalen

Admin Street2:

Admin Street3:

Admin City:Oslo

Admin State/Province:N/A

Admin Postal Code:N-0402

Admin Country:NO

Admin Phone:+1.4721604800

Admin Phone Ext.:

Admin FAX:+1.4721604801

Admin FAX Ext.:

Admin Email:hostmaster@trolltech.com

Tech ID:tuSfXVVJtggMdKWm

Tech Name:Trolltech AS

Tech Organization:Trolltech AS

Tech Street1:Sandakerveien 116, PO Box 4332 Nydalen

Tech Street2:

Tech Street3:

Tech City:Oslo

Tech State/Province:N/A

Tech Postal Code:N-0402

Tech Country:NO

…

f. Netstat

La commande **netstat** permet d’obtenir une foule d’informations sur le réseau et les protocoles.

Le paramètre **-i** permet d’obtenir l’état des cartes réseaux, afin de déterminer une éventuelle panne ou un problème de câble :

netstat -i

Table d’interfaces noyau

Iface MTU Met   RX-OK RX-ERR RX-DRP RX-OVR   TX-OK TX-ERR TX-DRP

TX-OVR Flg

eth0 1500   0 2332007   0   0   0 677842   0   0

0 BMRU

lo   16436   0   1109   0   0   0   1109   0   0

0 LRU

Si vous rajoutez le paramètre **-e**, vous obtenez le même résultat qu’avec ifconfig **-a**.

netstat -ei

Table d’interfaces noyau

eth0   Lien encap:Ethernet HWaddr 00:XX:D3:XX:AA:XX

inet adr:12.168.1.60 Bcast:192.168.1.255 Masque:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:2335314 errors:0 dropped:0 overruns:0 frame:0

TX packets:678095 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 lg file transmission:1000

RX bytes:1055212145 (1006.3 Mb) TX bytes:61264196 (58.4 Mb)

Interruption:20 Adresse de base:0x8c00

lo   Lien encap:Boucle locale

inet adr:127.0.0.1 Masque:255.0.0.0

UP LOOPBACK RUNNING MTU:16436 Metric:1

RX packets:1109 errors:0 dropped:0 overruns:0 frame:0

TX packets:1109 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 lg file transmission:0

RX bytes:60423 (59.0 Kb) TX bytes:60423 (59.0 Kb)

Le paramètre **-r** permet d'obtenir, comme route, les tables de routage. Ajoutez le paramètre **-n** pour indiquer les IPs à la place des noms.

netstat -rn

Table de routage IP du noyau

Destination   Passerelle   Genmask   Indic   MSS Fenêtre irtt

Iface

192.168.211.0   0.0.0.0   255.255.255.0   U   0 0   0

vmnet8

192.168.1.0   0.0.0.0   255.255.255.0   U   0 0   0

eth0

172.16.248.0   0.0.0.0   255.255.255.0   U   0 0   0

vmnet1

169.254.0.0   0.0.0.0   255.255.0.0   U   0 0   0

eth0

127.0.0.0   0.0.0.0   255.0.0.0   U   0 0   0

lo

0.0.0.0 192.168.1.1 0.0.0.0 UG 0 0 0

eth0

Le paramètre **-a** permet de visualiser toutes les connexions, pour tous les protocoles, y compris les ports en écoute de la machine. La sortie est trop longue pour être placée dans ces pages.

netstat -a | wc -l

495

Le paramètre **-A** permet de spécifier le protocole visible : inet, unix, ipx, ax25, netrom et ddp.

netstat -a -A inet

Connexions Internet actives (serveurs et établies)

Proto Recv-Q Send-Q Adresse locale Adresse distante

Etat

tcp 0 0 localhost:716 *:*

LISTEN

tcp 0 0 *:sunrpc *:*

LISTEN

tcp 0 0 localhost:ipp *:*

LISTEN

tcp 0 0 localhost:smtp *:*

LISTEN

tcp 0 0 localhost:hpssd *:*

LISTEN

tcp 0 0 slyserver:41851 imap.free.fr:imap

ESTABLISHED

tcp 0 0 slyserver:41850 imap.free.fr:imap

ESTABLISHED

tcp 0 0 slyserver:54220 by1msg4176111.gate:msnp

ESTABLISHED

tcp 0 0 slyserver:34267 by2msg2105007.phx.:msnp

ESTABLISHED

tcp 0 0 slyserver:47990 by1msg4082314.phx.:msnp

ESTABLISHED

udp 0 0 *:filenet-tms *:*

udp 0 0 *:mdns *:*

udp 0 0 *:sunrpc *:*

udp 0 0 *:ipp *:*

udp 0 0 172.16.248.1:ntp *:*

udp 0 0 192.168.211.1:ntp *:*

udp 0 0 slyserver:ntp *:*

udp 0 0 localhost:ntp *:*

udp 0 0 *:ntp *:*

raw 0 0 *:icmp *:* 7

Enfin le paramètre **-p** permet d'indiquer, quand c'est possible, le PID et le nom du processus

netstat -A inet -p

Connexions Internet actives (sans serveurs)

Proto Recv-Q Send-Q Adresse locale Adresse distante

Etat PID/Program name

…

tcp 0 0 slyserver:54220 by1msg4176111.gate:msnp

ESTABLISHED 4041/kopete

tcp 0 0 slyserver:34267 by2msg2105007.phx.:msnp

ESTABLISHED 4041/kopete

tcp 0 0 slyserver:47990 by1msg4082314.phx.:msnp

ESTABLISHED 4041/kopete

Fichiers généraux

a. /etc/resolv.conf

Le fichier **/etc/resolv.conf** est utilisé pour indiquer au système quels serveurs de noms et quels domaines interroger pour résoudre les requêtes DNS clientes. Les API sont incluses dans la bibliothèque et les API standards de Linux (il n'y a pas besoin d'ajouter des outils supplémentaires). On appelle cette bibliothèque le **resolver**.



En configurant DHCP ce fichier est en principe mis automatiquement à jour et ne devrait pas être modifié sauf si vous avez interdit la configuration DNS sur votre client.

$ cat /etc/resolv.conf

domain mondomaine.org

search mondomaine.org

nameserver 192.168.1.1

nameserver 192.168.1.2

domain : nom du domaine local. Les requêtes sont généralement réduites à des raccourcis relatifs au domaine local. S'il est absent le nom du domaine doit être déterminé à partir du nom d'hôte complet : c'est la partie située après le premier « . ».

search : liste des domaines de recherche. Par défaut lors de l'utilisation de raccourcis (noms d'hôtes courts) le resolver lance une recherche sur le domaine défini par la ligne domain, mais on peut spécifier ici une liste de domaines séparés par des espaces ou des virgules.

nameserver : adresse IP du serveur de noms (le serveur DNS). On peut en placer au maximum trois. Le resolver essaie d'utiliser le premier. En cas d'échec (timeout), il passe au second, et ainsi de suite.

options : des options peuvent être précisées. Par exemple **timeout:n** où n (en secondes) indique le délai d'attente de réponse d'un serveur de noms avant de passer au suivant.

b. /etc/hosts et /etc/networks

Sans même utiliser de serveur de noms, vous pouvez établir une correspondance entre les adresses IP et les noms des machines au sein du fichier **/etc/hosts**.

192.168.1.1 server1 www1 ftp

192.168.1.11 poste1

192.168.1.12 poste2

Vous pouvez faire de même pour nommer les réseaux (ce qui peut être utile pour les tcp_wrappers ou la commande route) dans le fichier **/etc/networks**.

loopnet 127.0.0.0

localnet 192.168.1.0

c. /etc/nsswitch.conf

Le fichier **/etc/nsswitch.conf** permet de déterminer l'ordre dans lequel le resolver (ou d'autres services) récupère ses informations. Les deux lignes en gras de l'exemple indiquent que lors d'une requête de résolution de nom (ou de réseau) les fichiers sont prioritaires. Le fichier **/etc/hosts** est d'abord lu, puis, si le resolver ne trouve pas l'information il passe par une résolution DNS.

passwd: compat

group: compat

hosts: files dns

networks: files dns

services: files

protocols: files

rpc: files

ethers: files

netmasks: files

netgroup: files nis

publickey: files

bootparams: files

automount: files nis

aliases: files



Il peut arriver que certains produits mal programmés n'utilisent pas le resolver mais directement le DNS, ou le fichier /etc/hosts, ou inversent l'ordre établi dans /etc/nsswitch.conf. Dans ce cas, il n'est pas possible de prévoir (et de prédire) le bon fonctionnement de ce genre de produits...

d. /etc/services

Le fichier **/etc/services** contient la liste des services réseaux connus de Unix ainsi que les ports et protocoles associés. Il est utilisé par de nombreux services (dont xinetd) et sous-systèmes comme le firewall de Linux.

Ce fichier est indicatif : c'est un fichier de description et de définition : tous les services de ce fichier ne tournent pas forcément sur votre machine (et heureusement vu le nombre) ! Et un service peut être configuré pour écouter un autre port. Par contre il est conseillé, lorsque vous rajoutez un service qui n'est pas présent dans ce fichier, de le rajouter à la fin.

tcpmux 1/tcp # TCP Port Service Multiplexer

tcpmux 1/udp # TCP Port Service Multiplexer

compressnet 2/tcp # Management Utility

compressnet 2/udp # Management Utility

compressnet 3/tcp # Compression Process

compressnet 3/udp # Compression Process

rje 5/tcp # Remote Job Entry

rje 5/udp # Remote Job Entry

echo 7/tcp Echo

echo 7/udp Echo

discard 9/tcp # Discard

discard 9/udp # Discard

systat 11/tcp users # Active Users

systat 11/udp users # Active Users

daytime 13/tcp # Daytime (RFC 867)

daytime 13/udp # Daytime (RFC 867)

netstat 15/tcp # Unassigned [was netstat]

qotd 17/tcp quote # Quote of the Day

qotd 17/udp quote # Quote of the Day

msp 18/tcp # Message Send Protocol

msp 18/udp # Message Send Protocol

chargen 19/tcp # Character Generator

chargen 19/udp # Character Generator

ftp-data 20/tcp # File Transfer [Default Data]

ftp-data 20/udp # File Transfer [Default Data]

ftp 21/tcp # File Transfer [Control]

fsp 21/udp # File Transfer [Control]

ssh 22/tcp # SSH Remote Login Protocol

ssh 22/udp # SSH Remote Login Protocol

telnet 23/tcp # Telnet

telnet 23/udp # Telnet

...

e. /etc/protocols

Le fichier **/etc/protocols** contient la liste des protocoles connus par Unix.

Assigned Internet Protocol Numbers

#

Decimal Keyword Protocol References

```
# -------  -------   --------            ----------
# protocol  num aliases    # comments
hopopt      0 HOPOPT     # IPv6 Hop-by-Hop Option      [RFC1883]
icmp        1 ICMP       # Internet Control Message      [RFC792]
igmp        2 IGMP       # Internet Group Management     [RFC1112]
ggp         3 GGP        # Gateway-to-Gateway           [RFC823]
ip          4 IP         # IP in IP (encapsulation)      [RFC2003]
st          5 ST         # Stream               [RFC1190,RFC1819]
tcp         6 TCP        # Transmission Control          [RFC793]
cbt         7 CBT        # CBT                     [Ballardie]
egp         8 EGP        # Exterior Gateway Protocol  [RFC888,DLM1]
igp         9 IGP        # any private interior gateway     [IANA]
bbn-rcc-mon  10 BBN-RCC-MON  # BBN RCC Monitoring              [SGC]
nvp-ii      11 NVP-II      # Network Voice Protocol      [RFC741,SC3]
pup         12 PUP         # PUP                     [PUP,XEROX]
argus       13 ARGUS       # ARGUS                          [RWS4]
emcon       14 EMCON       # EMCON                            [BN7]
xnet        15 XNET        # Cross Net Debugger         [IEN158,JFH2]
chaos       16 CHAOS       # Chaos                          [NC3]
udp         17 UDP         # User Datagram             [RFC768,JBP]
...
```

TP : Le réseau :

Configuration TCP/IP de Linux

Quelques commandes réseau

112.4 Configurer un client PPP (3)

PPP

Le protocole **PPP** (**Point to Point Protocol**) permet de vous relier à une autre machine afin d'y accéder, ou à son réseau, ce qui est souvent le cas d'Internet. Aujourd'hui encore et malgré les nombreuses solutions proposées par le câble ou l'ADSL, certaines connexions se font encore via un modem RTC (modem classique connecté sur port USB, série ou interne) relié à une prise téléphonique classique.

L'établissement d'une liaison PPP nécessite :

Un client disposant des outils ppp (pppd) et chat pour dialoguer avec le serveur.

Un serveur disposant de pppd et des moyens de fournir une adresse IP (dhcp).

Un modem.

La suite ne prend en considération que la partie cliente.

Vous devez connaître :

Le port série sur lequel est branché votre modem : ttySX (série ou USB), ttyACMX (usb), rfcommX (bluetooth), etc.

Le numéro d'appel de votre fournisseur d'accès Internet (FAI).

Le nom d'utilisateur et le mot de passe chez votre FAI.

L'adresse du serveur DNS de votre FAI.



Le modem n'est pas forcément RTC. Un téléphone portable reconnu comme modem via le câble de connexion au PC ou depuis le protocole Bluetooth fait un excellent modem. Pour peu qu'il soit aux normes 3G ou Edge, les débits peuvent être très impressionnants. Un grand nombre de connexions ADSL sont aussi effectuées via le protocole PPP. Dans ce cas la suite s'applique mais des modifications sont à prévoir.

3. Connexion via la console

a. À la main

Dans l'exemple qui suit :

Le numéro de téléphone est 0102030405.

Le login est « login ».

Le mot de passe est « password ».

Le périphérique est /dev/modem.

Il peut être nécessaire, bien que cela puisse être géré par DHCP au moment de la connexion, de modifier le fichier /etc/resolv.conf pour indiquer les serveurs de noms (DNS) de votre fournisseur.

La connexion PPP nécessite que le service **pppd** (généralement /usr/sbin/pppd) soit exécuté en tant que root. Pour cela, le droit SUID est souvent positionné :

-rwsr-xr-t 1 root root 316392 2008-04-04 19:03 pppd*

Une autre solution est de donner ces droits à l'outil de connexion (chat, kppp, etc.) ou de modifier en conséquence les droits des périphériques (cas de plusieurs distributions).

Les fichiers de configuration sont situés dans /etc/ppp :

ls -l /etc/ppp

total 48

-rw------- 1 root root 690 sep 21 2007 chap-secrets

-rw-r--r-- 1 root root 449 sep 21 2007 filters

lrwxrwxrwx 1 root root   5 mai 9 20:47 ip-down -> ip-up

drwxr-xr-x 2 root root 4096 sep 21 2007 ip-down.d

-rwxr-xr-x 1 root root 6175 avr 24 00:26 ip-up

drwxr-xr-x 2 root root 4096 sep 21 2007 ip-up.d

-rw-r--r-- 1 root root 7943 sep 21 2007 options

-rw------- 1 root root 340 sep 21 2007 options.pptp

-rw------- 1 root root 1219 sep 21 2007 pap-secrets

drwxr-xr-x 2 root root 4096 fév 23 23:18 peers

-rwxr-xr-x 1 root root 3778 avr 24 00:26 poll.tcpip

Voici un exemple de connexion à un serveur PPP :

#/bin/sh

/usr/sbin/pppd connect '/usr/sbin/chat -v ABORT ERROR ABORT "NO

CARRIER" \

ABORT BUSY "" ATZ OK ATDT0102030405 CONNECT "" ogin: "login" \

word: "password"' \

/dev/modem 38400 noipdefault debug crtscts modem defaultroute &

b. Par les fichiers

Vous allez avoir besoin de deux fichiers. Le premier va contenir les commandes du service pppd, le second la séquence de communication avec le FAI. Les deux sont placés dans /etc/ppp/peers.

Soit le premier fichier /etc/ppp/peers/cnx1 :

```
# cat /etc/ppp/peers/cnx1

/dev/modem

connect '/usr/sbin/chat -v -f /etc/ppp/peers/cnx1-chat'



defaultroute

noipdefault

usepeerdns

115200



debug

noauth



maxfail 10

lcp-echo-interval 5

lcp-echo-failure 12

holdoff 3

noaccomp noccp nobsdcomp nodeflate nopcomp novj novjccomp
```

lock

crtscts

Chaque ligne contient au moins une instruction dont voici les plus pertinentes :

/dev/modem : le périphérique de connexion (le modem) ;

connect : la chaîne de connexion envoyée au FAI ;

defaultroute : la route par défaut est remplacée par celle fournie par le FAI ;

noipdefault : le FAI fournit l'IP par son DHCP ;

usepeerdns : récupère les informations DNS du FAI ;

115200 : la vitesse de communication du périphérique (elle sera négociée) ;

debug : fournit le détail complet de la connexion ;

noauth : ce n'est pas le script ppp qui établit l'authentification (voir ligne connect) ;

maxfail : n tentatives de connexion avant d'abandonner ;

holdoff : attente de n secondes entre deux connexions ;

lock : permet l'accès exclusif au fichier périphérique ;

crtscts : active le contrôle de flux matériel.

Soit le second fichier /etc/ppp/peers/cnx1-chat utilisé par la ligne **connex** du premier fichier /etc/ppp/peers/cnx1 :

cat /etc/ppp/peers/cnx1-chat

ABORT ERROR

ABORT "NO CARRIER"

ABORT BUSY "" ATZ

OK ATDT0102030405

CONNECT ""

ogin: "login"

word: "password"

Il ne s'agit que d'un exemple. Vous devez vérifier tant du côté de votre FAI que du côté de la documentation de votre modem quelles sont les bonnes commandes AT à passer (elles sont généralement standard).

c. Connexion

Initialisez la connexion :

pppd call cnx1

Vous devriez voir les leds de votre modem clignoter, et si le haut-parleur est activé le bruit caractéristique se fait entendre. Si la connexion est établie, une nouvelle interface réseau apparaît : **ppp0**.

ifconfig ppp0

```
ppp0     Link encap:Point-Point Protocol
         inet addr:10.xx.yy.zz  P-t-P:10.xx.yy.zz Mask:255.255.255.0
         UP POINTOPOINT RUNNING  MTU:552  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0
         TX packets:0 errors:0 dropped:0 overruns:0
```

d. Connexion par front-end

Plutôt que d'établir une connexion depuis les commandes en ligne et des fichiers de configuration, il existe plusieurs outils graphiques, front-ends à PPP, qui permettent à la fois de configurer le modem et d'établir une connexion PPP avec tous les réglages réseaux possibles.

KDE est un très bon exemple avec l'outil **kppp**. L'interface principale ressemble à ce que vous avez peut-être pu déjà connaître sous Windows :

le choix d'un profil de connexion,

un nom d'utilisateur,

un mot de passe,

et les boutons, dont celui permettant de se connecter.



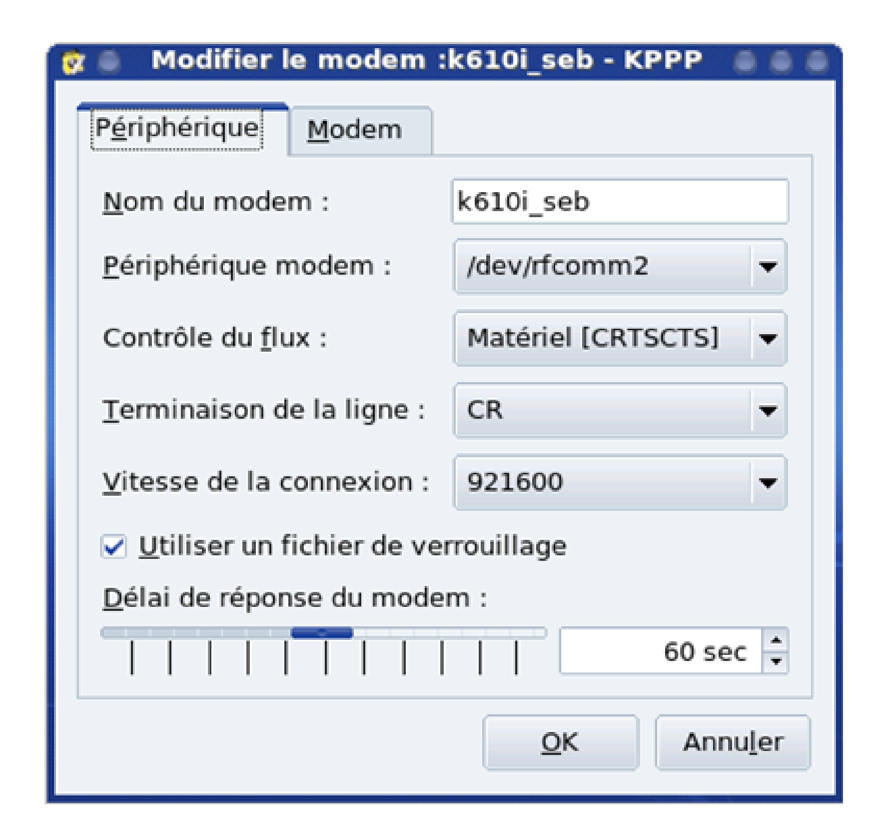
kppp permet d'établir une connexion PPP

La configuration se fait en trois étapes :

configuration du modem (la capture représente un téléphone portable 3G via une connexion bluetooth),

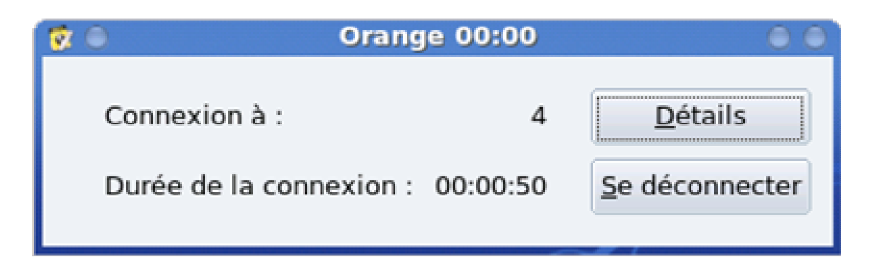
configuration des informations de connexion : profil, numéro de téléphone, sauvegarde du mot de passe, coût, etc.,

configuration des paramètres réseaux : réglages du DNS, du client DHCP, de la passerelle, etc.



Configuration du modem avec kppp

Enfin, une fois connecté une fenêtre d'état vous permet d'obtenir des informations détaillées sur la connexion, sa durée et vous offre la possibilité de vous déconnecter.



La liaison ppp est établie.



Les distributions sont souvent accompagnées d'outils de type NetworkManager qui permettent une connexion ethernet, Wi-Fi ou ppp à la volée en quelques clics. La distribution Mandriva mérite une mention spéciale : la configuration d'une connexion Internet via un mobile 3G au travers du protocole Bluetooth s'est fait en quelques secondes, l'outil Drakconf ayant tout détecté seul.

THEME : 113 Services réseau

113.1 Configurer xinetd et les services associés (4)

Services réseaux xinetd

1. Présentation

Le démon **xinetd** est un « super-service » permettant de contrôler l'accès à un ensemble de services, **telnet** par exemple. Beaucoup de services réseaux peuvent être configurés pour fonctionner avec xinetd, comme les services ftp, ssh, samba, rcp, http, etc. Des options de configuration spécifiques peuvent être appliquées pour chaque service géré.

Lorsqu'un hôte client se connecte à un service réseau contrôlé par xinetd, xinetd reçoit la requête et vérifie tout d'abord les autorisations d'accès TCP (voir **tcp_wrappers** au prochain chapitre) puis les règles définies pour ce service (autorisations spécifiques, ressources allouées, etc.). Une instance du service est alors démarrée et lui cède la connexion. À partir de ce moment **xinetd** n'interfère plus dans la connexion entre le client et le serveur.

2. Configuration

Les fichiers de configuration sont :

/etc/xinetd.conf : configuration globale

/etc/xinetd.d/* : répertoire contenant les fichiers spécifiques aux services. Il existe un fichier par service, du même nom que celui précisé dans /etc/services.

$ ls -l /etc/xinetd.d

total 92

-rw-r--r-- 1 root root 313 sep 22 2007 chargen

-rw-r--r-- 1 root root 333 sep 22 2007 chargen-udp

-rw-r--r-- 1 root root 256 mar 20 22:11 cups-lpd

-rw-r--r-- 1 root root 409 nov 4 2005 cvs

-rw-r--r-- 1 root root 313 sep 22 2007 daytime

-rw-r--r-- 1 root root 333 sep 22 2007 daytime-udp

-rw-r--r-- 1 root root 313 sep 22 2007 discard

-rw-r--r-- 1 root root 332 sep 22 2007 discard-udp

```
-rw-r--r-- 1 root root  305 sep 22  2007 echo
-rw-r--r-- 1 root root  324 sep 22  2007 echo-udp
-rw-r--r-- 1 root root  492 sep 22  2007 netstat
-rw-r--r-- 1 root root  207 avr 23 19:04 rsync
-rw-r--r-- 1 root root  337 fév 17 14:22 sane-port
-rw-r--r-- 1 root root  332 sep 22  2007 servers
-rw-r--r-- 1 root root  334 sep 22  2007 services
-rw-r--r-- 1 root root  351 jun 21  2007 svnserve
-rw-r--r-- 1 root root  277 nov  8  2007 swat
-rw-r--r-- 1 root root  536 sep 21  2007 systat
-rw-r--r-- 1 root root  387 fév  4 10:11 tftp.rpmsave
-rw-r--r-- 1 root root  339 sep 22  2007 time
-rw-r--r-- 1 root root  333 sep 22  2007 time-udp
-rw-r--r-- 1 root root 2304 avr  4 11:39 vnc
-rw------- 1 root root  768 sep 22  2007 vsftpd
```

Contenu de xinetd.conf :

```
defaults
{
        instances               = 60
        log_type                = SYSLOG authpriv
        log_on_success          = HOST PID
        log_on_failure          = HOST
        cps                     = 25 30
}
includedir /etc/xinetd.d
```

instances : nombre maximal de requêtes qu'un service xinetd peut gérer à un instant donné.

log_type : dans notre cas, les traces sont gérées par le démon **syslog** via **authpriv** et les traces sont placées dans /var/log/secure. **FILE /var/log/xinetd** aurait placé les traces dans /var/log/xinetd.

log_on_success : xinetd va journaliser l'événement si la connexion au service réussit. Les informations tracées sont l'hôte (**HOST**) et le **PID** du processus serveur traitant la connexion.

log_on_failure : idem mais pour les échecs. Il devient simple de savoir quels hôtes ont tenté de se connecter si par exemple la connexion n'est pas autorisée.

cps : xinetd n'autorise que 25 connexions par secondes à un service. Si la limite est atteinte, xinetd attendra 30 secondes avant d'autoriser à nouveau les connexions.

includedir : inclut les options des fichiers présents dans le répertoire indiqué.

Exemple /etc/xinetd.d/telnet :

```
# default: on

# description: The telnet server serves telnet sessions; it uses \

#       unencrypted username/password pairs for authentication.

service telnet

{

        disable = no

        flags           = REUSE

        socket_type     = stream

        wait            = no

        user            = root

        server          = /usr/sbin/in.telnetd

        log_on_failure  += USERID

}
```

La première ligne en commentaire, **default**, a une importance particulière. Elle n'est pas interprétée par xinetd mais par **ntsysv** ou **chkconfig** pour déterminer si le service est actif.

service : nom du service qui correspond à un service défini dans /etc/services.

flags : attributs pour la connexion. REUSE indique que la socket sera réutilisée pour une connexion telnet.

socket_type : spécifie le type de socket. Généralement **stream** (tcp) ou **dgram** (udp). Une connexion directe IP se fait par **raw**.

wait : indique si le serveur est single-threaded (yes) ou multi-threaded (no).

user : sous quel compte utilisateur le service sera lancé.

server : chemin de l'exécutable devant être lancé.

log_on_failure : le **+=** indique qu'on rajoute l'option associée au fichier de trace en plus de celles par défaut. Ici : le login.

disable : indique si le service est actif ou non.

Certaines options peuvent améliorer les conditions d'accès et la sécurité :

only_from : permet l'accès uniquement aux hôtes spécifiés.

no_access : empêche l'accès aux hôtes spécifiés (ex : 172.16.17.0/24).

access_times : autorise l'accès uniquement sur une plage horaire donnée (ex :09:00-18:30).

3. Démarrage et arrêt des services

On distingue deux cas.

Premier cas, le service **xinetd** est un service comme un autre dont le démarrage ou l'arrêt peut s'effectuer avec la commande **service** ou directement via l'exécution de /etc/init.d/xinetd.

```
# service xinetd start
```

Dans ce cas, la commande **chkconfig** (Red Hat, openSUSE) autorise ou non le lancement du service au démarrage pour chaque niveau d'exécution (runlevel).

```
# chkconfig --level 345 xinetd on
```

Second cas, comme xinetd gère plusieurs services, l'arrêt de xinetd arrête tous les services associés, et le démarrage de xinetd lance tous les services associés. Il n'est pas possible de choisir quels services de xinetd seront lancés dans tel ou tel niveau d'exécution. Mais vous pouvez choisir d'activer ou de désactiver simplement un service avec chkconfig.

```
# chkconfig telnet on
```

113.2 Configuration basique du MTA (4)

Courrier électronique

1. Principe

Quand un client (un utilisateur) envoie un message, il utilise un **MUA** (**Mail User Agent**), par exemple Outlook Express, Thunderbird, Evolution, Kmail, Mutt, etc.

Le **MUA** envoie le message au **MTA** (**Mail Transport Agent**). Le MTA étudie l'adresse électronique pour isoler l'utilisateur et le domaine de destination. Puis il vérifie les informations DNS de type **MX** (**Mail exchanger**) pour le domaine choisi, pour savoir à quel serveur transmettre le courrier. Si aucun MTA n'est disponible, le message est placé en file d'attente et relance la distribution plus tard (le délai dépend de la configuration du MTA).

Le MX peut être soit un autre MTA, qui jouera le rôle de routeur (cas d'une redirection vers un sous-domaine par exemple), soit un **MDA** (**Mail Delivery Agent**). Le MDA place le message dans un fichier temporaire, peut le filtrer, etc.

À ce niveau, le destinataire reçoit le message : soit il le récupère en lisant directement le fichier temporaire (cas de la commande mail par exemple) soit il passe par un protocole de type **POP** ou **IMAP**.

Le protocole de transport de messages est le **SMTP** (**Simple Mail Transfer Protocol**) sur le port 25.

Les protocoles de réception de messages soit **POP** (**Post Office Protocol**) sur le port 110 (POP3), soit **IMAP** (**Internet Message Access Protocol**).

Deux suites de courrier électronique se partagent l'essentiel du marché sur Unix : **sendmail** et **postfix**.

La suite libre **sendmail** est la plus connue et la plus utilisée. Sendmail a été créé en 1981 par Eric Allman et a été intégré à BSD 4.2 en 1983. On estimait en 2000 son utilisation à plus de 100 millions de serveurs de courrier électronique. Tant qu'il ne faut pas modifier fortement sa configuration de base, sendmail est idéal. Si vous souhaitez aller plus loin, l'achat d'un livre complet s'avère plus que nécessaire. La configuration de sendmail est si complexe qu'un langage de macros appelé **m4** a été inventé rien que pour lui. Aussi, vous n'éditez pas (ou très rarement) le fichier de configuration de sendmail : vous éditez le fichier source des macros et vous le « recompilez » : m4 va créer le fichier de configuration de sendmail. Sendmail est devenu un monstre de puissance et de configuration.

Le produit **postfix** tend à être de plus en plus utilisé non pas par les déçus de sendmail mais par ceux qui craignent de devoir le configurer. C'est une alternative à sendmail. Les buts de ses développeurs (dont certains sont ceux de sendmail) sont :

la compatibilité avec sendmail ;

la rapidité (plus de un million de messages par jour sur un simple Pentium 4) ;

la simplicité d'administration (fichier de configuration simple et lisible) ;

la sécurité (peut être chrootée) ;

la modularité (décomposition des traitements).

On retiendra la simplicité. En effet, on peut configurer postfix avec une seule commande sans avoir besoin d'éditer de fichier. C'est ce serveur que vous allez utiliser.

2. postfix

a. Configuration simple

La configuration de **postfix** est située dans /etc/postfix/main.cf. On modifie ses valeurs soit à la main, soit à l'aide de la commande **postconf**.

Postfix lance tout d'abord un service maître, **master**, qui sera chargé des processus secondaires **smtpd**, **pickup** et **nqmgr**.



Sur certaines distributions il faut modifier la configuration par défaut qui utilise la suite sendmail. Par exemple sur Red Hat vous devez indiquer d'utiliser postfix à la place de sendmail avec la commande **alternatives**.

#alternatives --set mta /usr/sbin/sendmail.postfix

Appliquez une configuration de base avec la commande **postconf**.

Domaine d'origine des messages

#postconf -e "myorigin = mondomaine.org"

De quels domaines recevoir le courrier

#postconf -e "mydestination = mondomaine.org"

De quels clients relayer le courrier

#postconf -e "mynetworks = 192.168.1.0/24, 127.0.0.1"




Sur quelles interfaces écouter

#postconf -e "inet_interface = all"

Lancez le service.

#service postfix start

ou :

#/etc/init.d/postfix start

b. Alias d'utilisateurs

Vous pouvez placer dans le fichier **/etc/aliases** des alias pour les utilisateurs locaux. Par exemple, si les messages de webmaster, admin et root doivent être redirigés vers regis :

regis: webmaster, admin, root

c. Test

Les traces sont placées dans /var/log/maillog. Testez le serveur de cette manière (par exemple) :

mail -s `echo $USER` root@server1 < /etc/passwd

Si tout fonctionne, vous obtiendrez des traces :

Fri 26 11:38:18 station1 postfix/pickup[12357] : F145040154: uid=0

from <root>

Fri 26 11:38:18 station1 postfix/cleanup[12318] : F145040154: mes-

sage-id=<20060126113017.F145040154:@station1.mondomaine.org>

Fri 26 11:38:26 station1 postfix/nqmgr[3469] : F145040154:

from=<root@station1.example.com>, size=314, nrcpt=1 (queue active)

Fri 26 11:38:32 station1 postfix/smtp[12468] : F145040154:

to=<root@server1>, relay=server1.mondomaine.org[192.168.1.1], de-

lay=17, status=sent (250 ok dirdel)

3. POP et IMAP

Il existe plusieurs suites pour gérer POP et IMAP. Une suite se nomme **cyrus-imap** et est en principe réservée aux grosses structures et aux serveurs 100% dédiés au courrier, c'est-à-dire où les utilisateurs ne se connectent pas.

Une solution se nomme **dovecot**. Après son installation, il suffit juste de le démarrer en tant que service pour que tout fonctionne, ou presque.

Éditez le fichier **/etc/dovecot.conf** pour vérifier les protocoles supportés.

protocols = imap pop3

Lancez le service :

service dovecot start

ou :

/etc/init.d/dovecot start

Testez en envoyant un message. Configurez un client de messagerie pour vérifier si le serveur POP fonctionne :

telnet localhost 110

trying 127.0.0.1...

Connected to localhost.localdomain.

Escape character is '^]'.

+OK POP3 localhost.localdomain server ready

USER regis

+OK user name accepted, password please

PASS password

+OK Mailbox open, 1 messages

STAT

+OK 1 384

TOP 1 99999

<message ici>

...

DELE 1

+OK Message deleted

QUIT

+OK bye

TP: Le réseau - Services réseau

113.3 Configuration basique de Apache (4)

Service HTTP Apache

1. Présentation

Apache 2 est le serveur HTTP le plus utilisé actuellement sur les serveurs Web. Sa configuration et sa flexibilité en font un serveur incontournable.

Lorsqu'un serveur Apache reçoit des requêtes, il peut les redistribuer à des processus fils. La configuration permet de lancer des processus de manière anticipée et d'adapter dynamiquement ce nombre en fonction de la charge.

Apache est modulaire. Chaque module permet d'ajouter des fonctionnalités au serveur. Le module le plus connu est probablement celui gérant le langage PHP, « mod_php ». Chaque module s'ajoute via les fichiers de configuration, et il n'y a même pas besoin de relancer le serveur Apache : on lui donne juste l'ordre de relire sa configuration.

Apache peut gérer plusieurs sites Web en même temps, ayant chacun leur nom, à l'aide des hôtes virtuels.

2. Arrêt/Relance

Le nom du service dépend de la distribution. Il est souvent intitulé **apache** ou **httpd**. Suivant la distribution lancez le service via la commande **service** ou directement par son nom /etc/init.d/apache.

/etc/init.d/httpd start : démarre ;

/etc/init.d/httpd stop : stoppe ;

/etc/init.d/httpd restart : redémarre ;

/etc/init.d/httpd reload : demande à Apache de relire sa configuration sans redémarrer.

Apache est fourni avec l'outil **apachectl** qui reprend les paramètres (liste non exhaustive) start, stop, status, reload, et surtout **configtest** qui valide ou non le contenu du fichier de configuration de Apache.

3. Configuration

La configuration principale est stockée dans /etc/httpd/conf/httpd.conf. Elle contrôle les paramètres généraux du serveur Web, les hôtes virtuels et les accès. La configuration des différents modules est placée dans /etc/httpd/conf.d. Les modules sont présents dans /etc/httpd/modules/. Par défaut la racine du serveur, celle où sont placées les pages du site, est dans /var/www ou /srv/www. Cette position dépend de la directive **DocumentRoot** dans les fichiers de configuration.

4. Directives générales

Il n'est pas possible de lister toutes les directives du fichier httpd.conf mais quelques-unes sont importantes.

ServerRoot : répertoire contenant les fichiers du serveur (configuration et modules). C'est généralement /etc/httpd.

Listen : ports sur lesquels le serveur Apache écoute. Par défaut 80 (443 en https). On peut en spécifier plusieurs avec plusieurs directives Listen. Si le serveur dispose de plusieurs adresses IP, on peut rajouter l'IP au port associé : **Listen 192.168.1.3:80**.

User : utilisateur des processus Apache. On n'utilise jamais root, mais un compte créé pour l'occasion, généralement Apache.

Group : idem mais pour le groupe.

ServerAdmin : adresse de courrier électronique de l'administrateur.

ServerName : nom d'hôte (et port) du serveur. Il ne correspond pas forcément au nom d'hôte de la machine. Par contre il doit être valide. **ServerName www.mondomaine.org**

UseCanonicalName : si elle vaut **on**, Apache va répondre en utilisant les informations de ServerName et Port, et pas les informations envoyées par le client. Par exemple, un http//192.168.1.3 se transforme en http://www.mondomaine.org.

UserDir : nom d'un sous-répertoire ou chaque utilisateur peut placer ses fichiers HTML personnels. Généralement public_html. On y accède avec **http://www.mondomaine.org/~login/page.html**.

ErrorLog : fichier où sont placées les logs d'erreur du serveur. /var/log/httpd/error_log.

CustomLog : fichier journal de Apache. **/var/log/httpd/access_log**.

Timeout : durée pendant laquelle le serveur attend des émissions/réceptions au cours d'une communication. Elle est réglée sur 300 secondes.

KeepAlive : définit si le serveur peut exécuter plus d'une requête par connexion. C'est à off par défaut mais si on passe à on Apache peut générer rapidement des processus enfants pour le soulager s'il est très chargé.

MaxKeepAliveRequests : nombre maximum de requêtes par connexion persistante. Une valeur élevée peut augmenter les performances du serveur. 100 par défaut.

KeepAliveTimeout : durée pendant laquelle le serveur (généralement un processus fils) attend après avoir servi une requête. Par défaut 15 secondes. Après les 15 secondes, la demande sera réceptionnée par le serveur avec un Timeout.

StartServers : nombre de serveurs créés au démarrage. Par défaut 8. Apache gérant ensuite dynamiquement le nombre de serveurs fils, ce paramètre a peu d'importance car le nombre va baisser ou augmenter rapidement.

5. Gestion des performances

MaxRequestPerChild : nombre de requêtes pouvant être exécutées par un processus fils avant de s'arrêter. Par défaut 4000. Ça permet une occupation mémoire plus réduite en la libérant plus rapidement.

MaxClients : limite du nombre total de requêtes pouvant être traitées simultanément. Par défaut 150. La valeur limite le risque de saturation du serveur.

MinSpareServers / MaxSpareServers : suivant la charge de la machine, Apache peut lancer d'autres processus serveurs pour s'adapter à la charge actuelle. Par défaut, elles sont de 5 et 20. Ces deux valeurs déterminent les nombres limites autorisés. Si un serveur est peu chargé avec 15 processus, Apache en supprimera mais en gardera toujours au moins 5. Si le serveur devient chargé avec 10 processus, Apache en créera des supplémentaires à concurrence de 20 maximum.

MinSpareThreads / MaxSpareThreads : chaque serveur fils peut accepter un certain nombre de requêtes simultanément. Pour cela il utilise les threads. Ces deux valeurs sont fixées par défaut à 20 et 75 et le mécanisme fonctionne comme ci-dessus.

ThreadsPerChild : nombre de threads par défaut au lancement d'un serveur fils. Par défaut 25.

6. Les répertoires, alias et emplacements

a. Directory

Les balises **<Directory chemin>** et **</Directory>** permettent de regrouper des directives qui ne s'appliqueront qu'au chemin (et à ses sous-répertoires) donnés. La directive **Options** est fortement conseillée.

```
<Directory /var/www/html/images>

   Options +Indexes +FollowSymLinks

   DirectoryIndex index.php index.html

   Order allow, deny

   Allow from All

</Directory>
```

<Directory /var/www/html/cgi-bin>

Options +ExecCGI

</Directory>

La directive **Options** accepte les valeurs suivantes précédées de + ou - et séparées par des espaces :

All : toutes les options sauf MultiViews ;

Indexes : si jamais le répertoire ne contient pas de fichier HTML par défaut (cf DirectoryIndex), le contenu du répertoire est affiché sous forme de listing ;

ExecCGI : l'exécution de scripts CGI est autorisée ;

FollowSymLinks : le serveur suit les liens symboliques.

La directive **DirectoryIndex** précise les fichiers html ou cgi par défaut lors du chargement d'une URL.

DirectoryIndex index.php index.html

Au chargement, sans préciser le nom du fichier html, le serveur tentera de charger index.php et, s'il est absent, index.html. Dans le cas contraire, c'est l'option Indexes qui détermine si le contenu doit être visible sous forme de répertoire.

La directive **Allow** indique quels clients seront autorisés à accéder au répertoire. Ce peut être **all**, un domaine, une IP, une IP tronquée (sous-réseau), une paire réseau/sous-réseau, etc. La directive **Deny** interdit l'accès et s'utilise de la même manière. L'ordre est déterminé par la directive **Order**.

b. Alias

La directive **Alias** permet de créer un raccourci entre l'arborescence logique du site Web et un chemin du système de fichiers.

Alias /help "/usr/share/doc/html"

Dans ce cas, l'url http://www.monsite.org/help ne cherchera pas dans le répertoire /var/www/html/help mais dans /usr/share/doc/html.

Contrairement aux balises **<Directory>**, les balises **<Location>** et **</Location>** permettent d'appliquer des directives basées sur l'URL (et pas les répertoires).

<Location /help>

Options +All -FollowSymLinks

Order deny, allow

Deny from all

Allow from .mondomaine.org

</Location>

7. Hôtes virtuels

Un serveur Apache est capable de gérer plusieurs sites Web sur un même serveur. Il existe plusieurs méthodes. La première se base sur les noms (plusieurs sites Web pour un serveur) l'autre sur les adresses ip (une adresse IP pour chaque site Web). Vous allez aborder la première version.

La directive **NameVirtualHost** spécifie l'adresse IP sur laquelle le serveur va recevoir les requêtes d'accès aux hôtes virtuels.

Les balises **<VirtualHost>** et **</Virtualhost>** permettent de définir un hôte virtuel.

NameVirtualHost 192.168.1.3

<VirtualHost 192.168.1.3>

ServerName www2.mondomaine.org

ServerAdmin webmaster@www2.mondomaine.org

DocumentRoot /var/www/www2.mondomaine.org/

ErrorLog logs/www2_error_log

CustomLog logs/www2_access_log

</VirtualHost>

Relancez Apache. Avec un navigateur (ex : Firefox) on vérifie si notre hôte virtuel répond avec l'URL http://www2.mondomaine.org (cette adresse doit être déclarée dans /etc/hosts ou connue du serveur de noms et pointer sur le bon serveur).

Remarquez cependant que si vous passez par http://www.mondomaine.org vous n'obtenez plus le site par défaut ! En effet quand on déclare des hôtes virtuels, le premier de la liste devient l'hôte par défaut et prioritaire.



Quand on accède au serveur, Apache recherche d'abord une correspondance entre le nom d'hôte spécifié par l'URL et chaque ServerName des hôtes virtuels. Si aucune correspondance exacte n'est trouvée, c'est le premier hôte virtuel qui est choisi par défaut en faisant abstraction des paramètres globaux.

Rajoutez un hôte virtuel pour le site principal.

```
<VirtualHost 192.168.1.3>
   ServerName        www.mondomaine.org
   ServerAdmin       webmaster@www.mondomaine.org
   DocumentRoot      /var/www/html
   ErrorLog      logs/error_log
   CustomLog      logs/access_log
</VirtualHost>
```

Attention, la règle ci-dessus s'applique aussi avec un nom court. Si vous inscrivez http://www ou http://www2 vous tomberez toujours sur l'hôte virtuel par défaut. Il faut demander http://www.mondomaine.org et http://www2.mondomaine.org.

On peut placer dans un hôte virtuel toutes les directives souhaitées (ou presque).

113.4 Gérer les partages NFS et SAMBA (4)

Partage de fichiers

1. NFS

a. Lancement

Le partage de fichier **NFS** (**Network File System**) ou système de fichiers réseau permet de partager tout ou partie de son système de fichiers à destination de clients NFS, bien souvent d'autres Unix. Dans sa version de base c'est un système simple et efficace. Nous allons étudier la version 2.

NFS s'appuie sur le **portmapper** (**portmap**), le support **nfs** du noyau et les services **rpc.nfsd** et **rpc.mountd**.

Pour lancer le service NFS, portmap et nfs doivent être lancés (en vérifier le statut avant).

service portmap status # /etc/init.d/portmap status ou rpcinfo -p

service nfs status

service portmap start

service nfs start

service nfslock start

Pour savoir si le service est disponible sur un hôte distant :

rpcinfo -p hote

b. Partage côté serveur

La liste des systèmes de fichiers à exporter se trouve dans **/etc/exports**. Il contient un partage par ligne.

```
# Rep exportes  Autorisations d'acces
/               poste1(rw) poste2(rw,no_root_squash)
/projects       *.mondomaine.org(rw)
/home/joe       poste*.mondomaine.org(rw)
/pub            192.168.1.0/255.255.255.0(ro)
```

Chaque ligne est composée de deux parties. La première est le chemin du répertoire exporté. La seconde contient les autorisations d'accès.

L'autorisation d'accès est composée de paires hôtes/permissions selon le format suivant :

host(permissions)

Si l'hôte n'est pas défini, c'est tout le réseau (portée dite mondiale) qui sera concerné par les permissions. Si les permissions ne sont pas définies, l'export sera en lecture seule. Il ne faut surtout pas mettre d'espaces entre l'hôte et les permissions. L'hôte peut être :

un nom d'hôte unique ;

un domaine ;

un réseau ou un sous-réseau ;

une combinaison de l'ensemble avec des caractères de substitution (*, ?).

Les permissions peuvent être :

ro : lecture seule ;

rw : lecture écriture ;

no_root_squash : le root distant équivaut au root local ;

root_squash : si root se connecte au partage, son uid sera remplacé par celui d'un utilisateur anonyme. Ainsi il n'y a pas de risques que l'utilisateur root d'un poste local puisse être root sur un partage distant ;

all_squash : étend la règle précédente à tous les utilisateurs ;

anonuid / anongid : uid et gid pour l'utilisateur anonyme.

Pour une gestion correcte des droits et des permissions, **les utilisateurs de même nom (login) doivent avoir les mêmes UID et GID sur le serveur et le client**. NFS se base en effet sur ces valeurs pour la sécurité des données du partage. Le nom de login seul ne suffit pas. Dans l'exemple ci-dessus, l'utilisateur **joe** est autorisé à accéder au partage **/home/joe** (on suppose que c'est son répertoire personnel) sur tous les postes du domaine. L'utilisateur joe doit être déclaré de la même manière (même UID) sur le serveur et sur tous les postes. C'est pour cela que l'on utilise souvent NIS avec NFS.

La commande **exportfs** permet de contrôler les partages.

exportfs -r : rafraîchit la liste des partages après modification de /etc/exports ;

exportfs -v : liste des partages ;

exportfs -a : exporte (ou recharge) tous les partages de /etc/exports ou un partage donné ;

exportfs -u : stoppe le partage donné. **-a** pour tous.

La commande **showmount** montre les partages d'un hôte donné.

showmount -e host

c. Montage côté client

Le support NFS est inclus sous forme de module du noyau. Il est automatiquement chargé à l'utilisation d'un accès NFS.

Dans **/etc/fstab** notez les modifications :

server1:/pub /mnt/pub nfs defaults 0 0

Le périphérique est remplacé par le chemin du partage sous la forme **serveur:chemin**. Le système de fichiers est **nfs**. C'est identique avec la commande **mount** :

mount -t nfs serveur1:/pub /mnt/pub



Si les montages NFS sont définis dans /etc/fstab, mount **-a** ne va pas les monter. Un service est généralement présent dans chaque distribution pour les monter et les démonter aux arrêts relances. Sur Red Hat, le service /etc/rc.d/init.d/netfs les montera automatiquement au démarrage.

NFS dispose d'options de montage spécifiques :

nolock : option de compatibilité avec d'anciens serveurs NFS ;

intr : interrompt une requête NFS si le serveur ne répond pas ;

hard : bloque les processus qui tentent d'accéder à un partage inaccessible ;

soft : un processus retournera une erreur en cas d'accès infructueux ;

rsize=8192, wsize=8192 : taille des blocs de lecture/écriture sur le serveur. Une écriture de 8 ko est plus rapide que 8 écritures de 1 ko.

FTP

Le serveur **FTP** (**File Transfer Protocol**) le plus courant est **vsftpd** (**Very Secure FTP Daemon**). Il a l'avantage d'être très petit, performant et rapide tout en étant tout de même très configurable (moins toutefois que Proftpd ou d'autres). Il convient dans la quasi-totalité des situations. C'est un service qui peut aussi bien être lancé par **xinetd** que (dans les dernières versions des distributions) en tant que service seul.

Deux niveaux de sécurité sont utilisables :

Anonyme : tout le monde peut se connecter au serveur FTP en tant que utilisateur **ftp** ou **anonymous**. l'environnement FTP est chrooté.

Utilisateur : les utilisateurs qui existent sur le serveur peuvent se connecter avec leur mot de passe et ont un accès complet à leurs données dans leur répertoire personnel.

Les utilisateurs anonymes étant considérés comme l'utilisateur ftp, c'est le répertoire personnel de ce compte qui est la racine du ftp.

Le fichier de configuration est présent dans /etc/vsftpd/vsftpd.conf.

La racine du ftp par défaut est dans /var/ftp.

Le script de lancement est /etc/init.d/vsftpd (service vsftpd start).

Pour activer ou non l'accès anonyme on modifie le fichier de configuration. Dans ce cas, l'utilisateur peut se connecter en tant que anonymous ou ftp. Dans tous les cas, il sera reconnu comme utilisateur « ftp » du serveur une fois connecté :

anonymous_enable=YES/NO

Pour activer ou non l’envoi de fichiers sur le serveur par des anonymes. Dans ce cas, l’autorisation d’écriture dans un répertoire est fonction des droits du répertoire sur le serveur (notamment si l’utilisateur ftp a le droit d’écrire ou non dans un répertoire) :

anon_upload_enable=YES/NO

Vous pouvez interdire à des utilisateurs de se connecter en plaçant leur noms dans /etc/vsftpd.ftpusers.

Vous pouvez ajouter des utilisateurs dans /etc/vsftpd.user_list si **userlist_enable=YES**. Dans ce cas, c’est la valeur de **userlist_deny (YES/NO)** qui déterminera si le fichier contient les utilisateurs interdits ou autorisés.

On peut créer dans chaque répertoire du serveur un fichier **.message**. Dans ce cas, son contenu sera affiché lors de l’accès au répertoire.

TP : Le réseau - Partages de fichiers

113.5 Gérer des noms de domaines DNS de base (4)

Serveur DNS

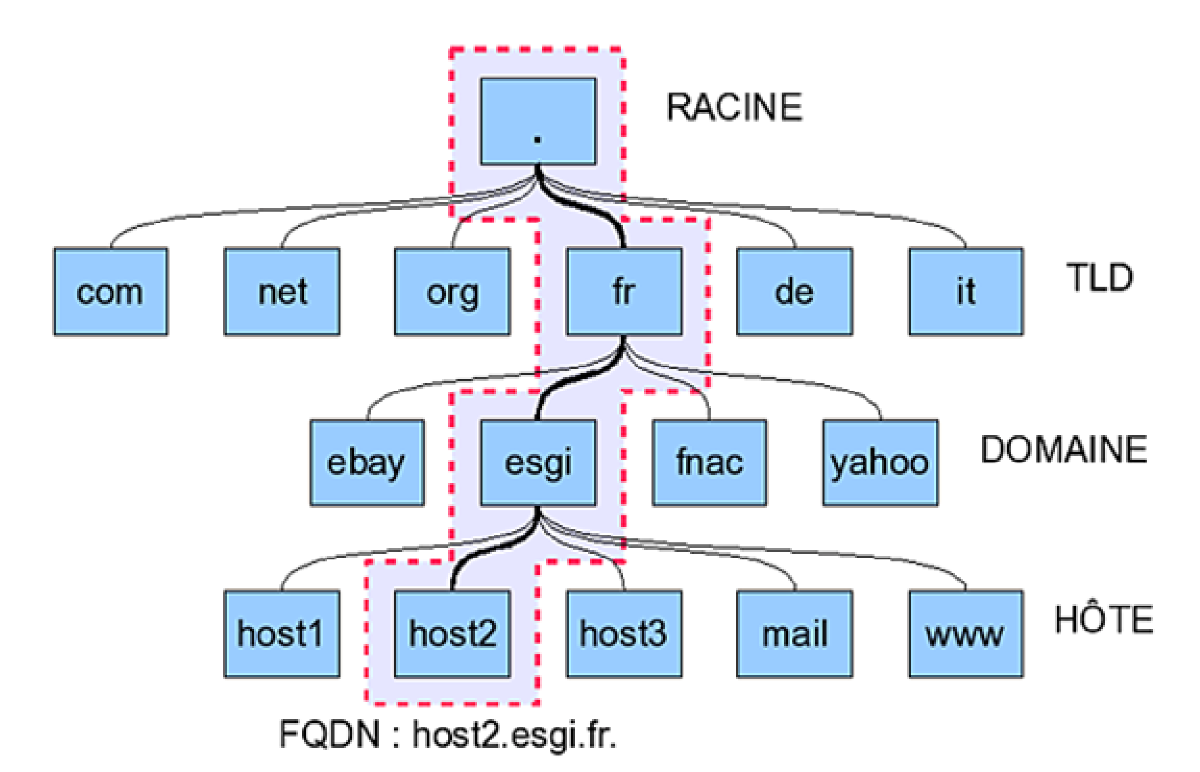
1. Présentation

Le Système de Noms de Domaines **DNS** (**Domain Name System**) transforme les noms d’hôte en adresses IP : c’est la **résolution de nom**. Il transforme les adresses IP en noms d’hôte : c’est la **résolution inverse**. Il permet de regrouper les machines par domaines de nom. Il fournit des informations de routage et de courrier électronique.

Le DNS permet de faire référence à des systèmes basés sur IP (les **hôtes**) à l’aide de noms conviviaux (les **noms de domaines**). L’intérêt d’un DNS est évident. Les noms de domaine sont plus simples à retenir, et si son adresse IP change l’utilisateur ne s’en rend même pas compte. On comprend que le DNS est un service clé critique pour Internet.

Les noms de domaine sont séparés par des points, chaque élément pouvant être composé de 63 caractères ; il ne peut y avoir qu’un maximum de 127 éléments et le nom complet ne doit pas dépasser 255 caractères. Le nom complet non abrégé est appelé **FQDN** (**Fully Qualified Domain Name**). Dans un FQDN, l’élément le plus à droite est appelé **TLD (Top Level Domain**), celui le plus à gauche représente l’hôte et donc l’adresse IP.

Le DNS contient une configuration spéciale pour les routeurs de courrier électronique (définitions MX) permettant une résolution inverse, un facteur de priorité et une tolérance de panne.



Représentation d'une arborescence DNS

Une zone est une partie d'un domaine gérée par un serveur particulier. Une zone peut gérer un ou plusieurs sous-domaines, et un sous-domaine peut être réparti en plusieurs zones. Une zone représente l'unité d'administration dont une personne peut être responsable.

2. Lancement

Le service s'appelle **named**.

service named start

Ou :

/etc/init.d/named start

3. Configuration de Bind

Bind (**Berkeley Internet Name Daemon**) est le serveur de noms le plus utilisé sur Internet. Bind 9 supporte l'IPv6, les noms de domaine unicode, le multithread et de nombreuses améliorations de sécurité.

a. Configuration générale

La configuration globale de Bind est placée dans le fichier **/etc/named.conf**. La configuration détaillée des zones est placée dans /var/lib/named. /etc/named.conf est composé de deux parties. La première concerne la configuration globale des options de Bind. La

seconde est la déclaration des zones pour les domaines individuels. Les commentaires commencent par un # ou //.



Attention il arrive parfois (notamment sur RHEL 4.x) que la configuration de Bind soit « chrootée » (déplacée dans une arborescence spécifique d'où le service ne peut sortir, le reste de l'arborescence lui étant inaccessible). Sur Centos et RHEL 4.x et supérieurs named.conf est dans /var/named/chroot/etc/.On peut modifier ce mode en modifiant le fichier de configuration /etc/sysconfig/named.

```
# cat /etc/sysconfig/named

...

CHROOT=/var/named/chroot

...
```

Dans ce cas, tous les fichiers de configuration, y compris les zones, sont relatifs à ce chemin. Voici un fichier named.conf de base.

```
options {

   directory "/var/lib/named";

   forwarders { 10.0.0.1; };

   notify no;

};

zone "localhost" in {

   type master;

   file "localhost.zone";

};

zone "0.0.127.in-addr.arpa" in {

   type master;

   file "127.0.0.zone";

};

zone "." in {
```

type hint;

file "root.hint";

};

b. Section globale

La configuration globale est placée dans la section **options**. Voici un détail de quelques options importantes (le point-virgule doit être précisé) :

directory "filename"; : emplacement des fichiers contenant les données des zones.

forwarders { adresse-ip; }; : si le serveur bind ne peut résoudre lui-même la requête, elle est renvoyée à un serveur DNS extérieur, par exemple celui du fournisseur d'accès.

listen-on-port 53 {127.0.0.1; adresse-ip; }; : port d'écoute du DNS suivi des adresses d'écoute. On indique ici les adresses IP des interfaces réseau de la machine. Il ne faut pas oublier 127.0.0.1.

allow-query { 127.0.0.1; réseau; }; : machine(s) ou réseau(x) autorisés à utiliser le service DNS. Par exemple 192.168.1/24. Si la directive est absente, tout est autorisé.

allow-transfer { 192.168.1.2; }; : machine(s) ou réseau(x) autorisés à copier la base de données dans le cas d'une relation maître et esclave. Par défaut aucune copie n'est autorisée.

notify no; : on notifie ou non les autres serveurs DNS d'un changement dans les zones ou d'un redémarrage du serveur.

c. Section de zones

Pour chaque domaine ou sous-domaine, on définit deux sections **zone**. La première contient les informations de résolution de nom (nom vers IP) et la seconde les informations de résolution inverse (IP vers Nom). Dans chacun des cas, la zone peut être maître **Master** ou esclave **Slave** :

Master : le serveur contient la totalité des enregistrements de la zone dans ses fichiers de zone. Lorsqu'il reçoit une requête, il cherche dans ses fichiers (ou dans son cache) la résolution de celle-ci.

Slave : le serveur ne contient par défaut aucun enregistrement. Il se synchronise avec un serveur maître duquel il récupère toutes les informations de zone. Ces informations peuvent être placées dans un fichier. Dans ce cas l'esclave stocke une copie locale de la base. Lors de la synchronisation, le numéro de série de cette copie est comparé à celui du maître. Si les numéros sont différents, une nouvelle copie a lieu, sinon la précédente continue à être utilisée.

d. Zone de résolution

Elle est généralement appelée **zone**. Pour chaque domaine ou sous-domaine, elle indique dans quel fichier sont placées les informations de la zone (c'est-à-dire et entre autres les adresses IP associées à chaque hôte), son type (maître ou esclave), si on autorise ou non la notification, l'adresse IP du serveur DNS maître dans le cas d'un esclave, etc.

Le nom de la zone est très important puisque c'est lui qui détermine le domaine de recherche. Quand le DNS reçoit une requête, il recherche dans toutes les zones une correspondance.

```
zone "domaine.org" {

   type    "master";

   file   "domaine.org.zone";

};
```

type : master ou slave ;

file : nom du fichier qui contient les informations de la zone. Il n'y a pas de règles précises de nommage mais pour des raisons de lisibilité il est conseillé de lui donner le même nom que la zone tant pour une zone master que pour une slave. Pour un master, c'est l'original éventuellement rempli par vos soins. Pour un slave, ce n'est pas obligatoire. S'il est présent, ce sera une copie du master, synchronisée.

Dans le cas d'un Master, on peut rajouter **allow-transfer** (serveurs autorisés à dupliquer la zone) et **notify yes** (indique une mise à jour ou une relance pour les slaves).

En cas de Slave : on rajoute la directive **masters** pour indiquer à partir de quel serveur Master dupliquer.

e. Zone de résolution inverse

Pour chaque réseau ou sous-réseau IP (ou plage d'adresses) on définit une zone de résolution inverse dont le fichier contient une association IP vers nom de machine. C'est en fait presque la même chose que la zone de résolution sauf que l'on doit respecter une convention de nommage :

Le nom de la zone se termine toujours par une domaine spécial .**in-addr.arpa**.

On doit tout d'abord déterminer quel réseau la zone doit couvrir (cas des sous-réseaux). Pour nous : un réseau de classe C 192.168.1.0 soit **192.168.1/24**.

On inverse l'ordre des octets dans l'adresse : **1.168.192**.

On ajoute **in-addr.arpa**. Notre nom de zone sera donc **1.168.192.in-addr.arpa**.

Pour le reste, les mêmes remarques que pour la zone de résolution s'appliquent.

```
Zone "1.168.192.in-addr.arpa" {
    type        master;
    file       "192.168.1.zone";
};
```

f. Exemple

Soit un domaine **domaine.org** sur un réseau de classe C 192.168.1.0. Soit deux serveurs DNS 192.168.1.1 Master et 192.168.1.2 Slave.

Sur le Master

```
zone "domaine.org" {
    type    master;
    file    "domaine.org.zone";
    allow-transfer { 192.168.1.2; } ;
    notify yes;
};
zone "1.168.192.in-addr.arpa" {
    type    master;
    file    "192.168.1.zone";
    allow-transfer { 192.168.1.2; } ;
    notify yes;
};
```

Sur le Slave

```
zone "domaine.org" {
    type       slave;
    file       "domaine.org.zone";
    masters     { 192.168.1.1; };
```

```
};

zone "1.168.192.in-addr.arpa" {

   type       slave;

   file       "192.168.1.zone";

   masters    { 192.168.1.1; } ;

};
```

g. Zones spéciales

La zone racine « . » permet de spécifier les serveurs racines. Quand aucune des zones n'arrive à résoudre une requête, c'est la zone racine qui est utilisée par défaut et qui renvoie sur les serveurs racines.

La zone de loopback n'est pas nécessaire bien que utile. Elle fait office de **cache DNS**. Quand une requête arrive sur le serveur et qu'il ne possède pas l'information de résolution, il va la demander aux serveurs DNS racines qui redescendront l'information. Celle-ci est alors placée en cache. Du coup les accès suivants seront bien plus rapides !

4. Fichiers de zones

a. Définitions

Les fichiers de zones utilisent plusieurs termes, caractères et abréviations spécifiques.

RR : **Ressource Record**. Nom d'un enregistrement DNS (les données du DNS).

SOA : **Star Of Authority**. Permet de décrire la zone.

IN : **the Internet**. Définit une classe d'enregistrement qui correspond aux données Internet (IP). C'est celle par défaut si elle n'est pas précisée pour les enregistrements.

A : **Address**. Permet d'associer une adresse IP à un nom d'hôte. Pour Ipv6 c'est AAAA.

NS : **Name Server**. Désigne un serveur DNS de la zone.

MX : **Mail eXchanger**. Désigne un serveur de courrier électronique, avec un indicateur de priorité. Plus la valeur est faible, plus la priorité est élevée.

CNAME : **Canonical Name**. Permet de rajouter des alias : lier un nom à un autre. On peut créer des alias sur des noms d'hôte et aussi sur des alias.

PTR : **Pointer**. Dans une zone de résolution inverse, fait pointer une IP sur un nom d'hôte.

TTL : **Time To Live**. Durée de vie des enregistrements de la zone.

@ : dans les déclarations de la zone, c'est un alias (caractère de remplacement) pour le nom de la zone déclarée dans /etc/named.conf. Ainsi si la zone s'appelle domaine.org, @ vaut domaine.org. Dans la déclaration de l'administrateur de la SOA, il remplace ponctuellement le point dans l'adresse de courrier électronique.

Le point « **.** »: Si l'on omet le point en fin de déclaration d'hôte, le nom de la zone est concaténé à la fin du nom. Par exemple pour la zone domaine.org, si on écrit **poste1**, cela équivaut à **poste1.domaine.org**. Si on écrit **poste1.domaine.org** (sans le point à la fin) alors on obtient comme résultat **poste1.domaine.org.domaine.org** ! Pour éviter cela, vous devez écrire **poste1.domaine.org.** (notez le point à la fin).

Certains enregistrements nécessitent une notion de durée, qui est généralement exprimée en secondes, mais aussi parfois avec des abréviations :

1M : une minute, soit 60 secondes (1M, 10M, 30M, etc.) ;

1H : une heure, 3600 secondes ;

1D : un jour, 86400 secondes ;

1W : une semaine, 604800 secondes ;

365D : un an, 31536000 secondes.



Attention, et ceci est très important : dans les fichiers de zones, IL NE FAUT JAMAIS COMMENCER UNE LIGNE PAR DES ESPACES OU TABULATIONS. Ça ne marche absolument pas : les espaces ou tabulations seraient interprétés comme faisant partie du nom indiqué, de l'adresse ou de l'option.

b. Zone

Commencez tout d'abord par une directive **TTL** qui indique le temps de vie de la zone en secondes. Cela signifie que chaque enregistrement de la zone sera valable durant le temps indiqué par **TTL** (note : il est possible de modifier cette valeur pour chaque enregistrement). Durant ce temps, les données peuvent être placées en cache par les autres serveurs de noms distants. Une valeur élevée permet de réduire le nombre de requêtes effectuées et de rallonger les délais entre les synchronisations.

$TTL 86400

Après les directives TTL, placez un enregistrement de ressources **SOA** :

<domain> IN SOA <primary-name-server> <hostmaster-email> (

<serial-number>

<time-to-refresh>

<time-to-retry>

<time-to-expire>

<minimum-TTL>)

domain : c'est le nom de la zone, le même nom que celui utilisé dans /etc/named.conf. On peut le remplacer par @ sinon il ne faut pas oublier de le terminer par un point (pour éviter une concaténation).

primary-name-server : le nom sur le serveur DNS maître sur cette zone. Il ne faudra pas oublier de le déclarer dans la liste des hôtes (enregistrements PTR ou A).

hostmaster-email : adresse de courrier électronique de l'administrateur du serveur de nom. Le caractère @ étant déjà réservé à un autre usage, on utilise un point pour le remplacer. Ainsi « admin@domaine.org » devra s'écrire « **admin.domaine.org.** » .

serial-number : c'est un numéro de série que l'on doit incrémenter manuellement à chaque modification du fichier zone pour que le serveur de nom sache qu'il doit recharger cette zone. Elle est utilisée pour la synchronisation avec les serveurs esclaves. Si le numéro de série est le même qu'à la dernière synchronisation les données ne sont pas rafraîchies. Par convention on place **YYYYMMDDNN** (année-mois-jour-numéro) sur dix chiffres.

time-to-refresh : indique à tout serveur esclave combien de temps il doit attendre avant de demander au serveur de noms maître si des changements ont été effectués dans la zone.

time-to-retry : indique au serveur esclave combien de temps attendre avant d'émettre à nouveau une demande de rafraîchissement si le serveur maître n'a pas répondu. La demande aura lieu toutes les time-to-retry secondes.

time-to-expire : si malgré les tentatives de contacts toutes les time-to-retry secondes le serveur n'a pas répondu au bout de la durée indiquée dans time-to-expire, le serveur esclave cesse de répondre aux requêtes pour cette zone.

Minimum-TTL : le serveur de nom demande aux autres serveurs de noms de mettre en cache les informations pour cette zone pendant au moins la durée indiquée.

```
@   IN   SOA   dns1.domaine.org.   hostmaster.domaine.org. (

        2005122701    ; serial

        21600         ; refresh de 6 heures

        3600          ; tenter toutes les 1 heures
```

604800 ; tentatives expirent après une semaine

86400) ; TTL mini d'un jour

Passez ensuite au enregistrements **NS** (**Name Server**) où vous spécifiez les serveurs de noms de cette zone.

IN NS dns1

IN NS dns2



Quand on ne spécifie pas en début de ligne un nom d'hôte ou de zone (complet ou @), cela veut dire qu'on utilise le même que la ligne du dessus. Tant qu'on n'en précise pas de nouveau, c'est le dernier indiqué qui est utilisé. Ainsi ci-dessus les lignes pourraient être :

@ IN NS dns1

@ IN NS dns2

ou :

domaine.org. IN NS dns1

domaine.org. IN NS dns2



Notez l'absence de point après le nom de l'hôte et donc domaine.org est concaténé pour obtenir dns1.domaine.org.

IN NS dns1

équivaut à :

IN NS dns1.domaine.org.

Passez ensuite à l'énumération des serveurs de courrier électronique de la zone. La valeur numérique située après MX indique la priorité. Plus la valeur est basse plus le serveur est prioritaire et susceptible d'être contacté en premier. Si les valeurs sont identiques, le courrier est redistribué de manière homogène entre les serveurs. Si un serveur ne répond pas (chargé, en panne) la bascule vers une autre machine est automatique.

IN MX 10 mail

IN MX 15 mail2

Si vous souhaitez qu'une machine réponde en passant par le FQDN domaine.org sans préciser d'hôte (par exemple http://domaine.org sans utiliser http://www.domaine.org) alors vous pouvez maintenant déclarer une adresse IP pour ce serveur. Ainsi la commande **ping domaine.org** répondra 192.168.1.3 !

IN A 192.168.1.3

Vous pouvez maintenant déclarer les autres hôtes dont les serveurs de noms, de mails, les postes, etc.

dns1 IN A 192.168.1.1

dns2 IN A 192.168.1.2

server1 IN A 192.168.1.3

server2 IN A 192.168.1.4

poste1 IN A 192.168.1.11

poste2 IN A 192.168.1.12

poste3 IN A 192.168.1.13

On remarque que nos serveurs mail et mail2 ne sont pas déclarés, et que l'on n'a pas indiqué de serveur Web et ftp. Nous allons utiliser les alias, en faisant pointer ces noms d'hôtes sur d'autres hôtes.

mail IN CNAME server1

mail2 IN CNAME server2

www IN CNAME server1

ftp IN CNAME server1

La configuration de la zone est terminée, il faut maintenant s'occuper de la zone de résolution inverse.

c. Zone de résolution inverse

La zone de révolution inverse est presque identique à la précédente, si ce n'est que les enregistrements A sont remplacés par des enregistrements PTR destinés à traduire une IP en hôte. Le TTL et la déclaration SOA doivent être si possible identiques (sauf le nom de la zone). Vous placez aussi les enregistrements NS.

IN NS dns1.domaine.org.

IN NS dns2.domaine.org.

Vous n’êtes pas obligé de placer dans la zone de résolution inverse la traduction des adresses IP du DNS, étant donné que c’est le DNS lui-même qui résout son propre nom ! Cependant le faire peut accélérer la démarche, le DNS n’ayant pas à exécuter une requête sur lui-même. Passez aux enregistrements PTR traduisant l’adresse IP pour chaque hôte.

1 IN PTR dns1.domaine.org.

2 IN PTR dns2.domaine.org.

3 IN PTR server1.domaine.org.

4 IN PTR server2.domaine.org.

11 IN PTR poste1.domaine.org.

12 IN PTR poste2.domaine.org.

13 IN PTR poste3.domaine.org.

Il est théoriquement possible pour la même IP d’attribuer plusieurs hôtes ; les RFC ne sont pas très explicites sur cette possibilité qui, au final, peut créer des problèmes.

5. Diagnostic des problèmes de configuration

La commande **named-checkconf** vérifie la syntaxe du fichier **named.conf**. Vous lui fournissez en paramètre le fichier. La sortie indiquera les lignes posant problème.

La commande **named-checkzone** vérifie la syntaxe d’un fichier de zone (y compris de résolution inverse). Vous lui spécifiez en paramètre le nom du fichier zone.

a. Interrogation dig et host

Le programme **dig** est un outil d’interrogation avancé de serveur de noms, capable de restituer toutes les informations des zones.

> dig free.fr

; <<>> DiG 9.4.1-P1 <<>> free.fr

;; global options: printcmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63972

;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

```
;; QUESTION SECTION:
;free.fr.                IN      A


;; ANSWER SECTION:
free.fr.          86363   IN      A       212.27.48.10


;; Query time: 1 msec
;; SERVER: 10.23.254.240#53(10.23.254.240)
;; WHEN: Wed May 14 09:36:09 2008
;; MSG SIZE  rcvd: 41
```

Par défaut dig ne restitue que l'adresse de l'hôte passé en paramètre. En cas de réussite, le statut vaut **NOERROR**, le nombre de réponses est indiqué par **ANSWER** et la réponse se situe en dessous de la section **ANSWER**. Pour obtenir une résolution inverse il existe deux solutions.

```
$ dig 10.48.27.212.in-addr.arpa ptr
```

ou plus simplement :

```
$ dig -x 212.27.48.10


; <<>> DiG 9.4.1-P1 <<>> -x 212.27.48.10
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60222
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0


;; QUESTION SECTION:
```

;10.48.27.212.in-addr.arpa. IN PTR

;; ANSWER SECTION:

10.48.27.212.in-addr.arpa. 86400 IN PTR www.free.fr.

;; Query time: 31 msec

;; SERVER: 10.23.254.240#53(10.23.254.240)

;; WHEN: Wed May 14 09:36:51 2008

;; MSG SIZE rcvd: 68

Dans la première syntaxe, remarquez que vous pouvez rajouter un paramètre d'interrogation. Voici les principaux.

a : uniquement l'adresse ;

any : toutes les informations concernant le domaine ;

mx : les serveurs de messagerie ;

ns : les serveurs de noms ;

soa : la zone Start of Authority ;

hinfo : infos sur l'hôte ;

txt : texte de description ;

ptr : zone reverse de l'hôte ;

axfr : liste de tous les hôtes de la zone.

$ dig free.fr any

; <<>> DiG 9.4.1-P1 <<>> free.fr any

;; global options: printcmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28893

```
;; flags: qr aa; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 8

;; QUESTION SECTION:
;free.fr.                IN      ANY

;; ANSWER SECTION:
free.fr.         86400   IN      NS      freens2-g20.free.fr.
free.fr.         86400   IN      A       212.27.48.10
free.fr.         86400   IN      NS      freens1-g20.free.fr.
free.fr.         86400   IN      MX      20 mx2.free.fr.
free.fr.         86400   IN      SOA     freens1-g20.free.fr.
hostmaster.proxad.net. 2008051001 10800 3600 604800 86400
free.fr.         86400   IN      MX      10 mx1.free.fr.

;; ADDITIONAL SECTION:
freens2-g20.free.fr.   86400   IN      A       212.27.60.20
mx1.free.fr.           86400   IN      A       212.27.48.6
mx2.free.fr.           86400   IN      A       212.27.42.56
freens1-g20.free.fr.   86400   IN      A       212.27.60.19
mx2.free.fr.           86400   IN      A       212.27.42.58
mx1.free.fr.           86400   IN      A       212.27.48.7
mx2.free.fr.           86400   IN      A       212.27.42.57
mx2.free.fr.           86400   IN      A       212.27.42.59

;; Query time: 9 msec
```

;; SERVER: 10.23.254.240#53(10.23.254.240)

;; WHEN: Wed May 14 09:35:32 2008

;; MSG SIZE rcvd: 318

L'outil **host** fournit le même résultat de manière peut-être un peu plus simple.

$ host free.fr

free.fr has address 212.27.48.10

free.fr mail is handled by 10 mx1.free.fr.

$ host -t any free.fr

free.fr has address 212.27.48.10

free.fr name server freens1-g20.free.fr.

free.fr has SOA record freens1-g20.free.fr. hostmaster.proxad.net.

2008051001 10800 3600 604800 86400

free.fr mail is handled by 10 mx1.free.fr.

$ host -a free.fr

Trying "free.fr"

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64513

;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 2

;; QUESTION SECTION:

;free.fr.   IN   ANY

;; ANSWER SECTION:

free.fr.   86140   IN   A   212.27.48.10

free.fr. 86140 IN NS freens1-g20.free.fr.

free.fr. 86140 IN SOA freens1-g20.free.fr.

hostmaster.proxad.net. 2008051001 10800 3600 604800 86400

free.fr. 86140 IN MX 10 mx1.free.fr.

;; ADDITIONAL SECTION:

freens1-g20.free.fr. 86140 IN A 212.27.60.19

mx1.free.fr. 86140 IN A 212.27.48.7

Received 176 bytes from 10.23.254.240#53 in 4 ms

TP : Le réseau - Le resolver

113.7 Paramétrer un shell sécurisé SSH de base (4)

Voire 112.4 Configurer un client PPP (3)

THEME : 114 Sécurité

114.1 Tâches d'administration de sécurité (4)

Bases de sécurité

1. Sécurité informatique

Les objectifs principaux de la sécurité informatique concernent :

La sécurité de la connexion : il s'agit de contrôler que les utilisateurs qui se connectent sont bien autorisés à le faire et de leur interdire l'accès au système dans le cas contraire.

L'intégrité des données : il s'agit de faire en sorte que les fichiers et les bases de données ne soient pas corrompus et de maintenir la cohérence entre les données.

La confidentialité des données : l'accès aux données en consultation et en modification doit être limité aux seuls utilisateurs autorisés.

Vous disposez de plusieurs moyens :

L'authentification des utilisateurs par un mot de passe.

Le cryptage des données.

La sécurité physique en contrôlant l'accès des personnes aux salles informatiques, en utilisant des circuits inviolables matériellement.

L'information sur les risques pénaux encourus en cas d'infraction. Un « braquage » informatique est un délit, pas un jeu.

Le contrôle fréquent des droits d'accès aux fichiers et aux bases de données.

Le contrôle des « checksum » des fichiers pour s'assurer de leur intégrité.

La sauvegarde régulière des données.

L'audit des principaux événements du système.

L'installation de murs de feu « firewall » qui contrôlent les accès au système informatique depuis l'extérieur et limitent l'accès à des services externes par des utilisateurs non avertis ou qui n'en ont pas besoin pour, par exemple, limiter le risque de rapatriement de virus.

L'installation d'un antivirus, même sous Linux, si le serveur traite des données depuis et vers des systèmes d'exploitation concernés par les virus.

L'installation d'outils anti-spams et anti-spywares, selon le même principe, afin d'éviter une intrusion et la saturation des serveurs de courrier électronique.

Le démarrage uniquement des services réellement utiles sur le serveur et sur le client.

Quelques méthodes simples permettent de limiter les risques :

Vous pouvez définir une valeur de umask restrictive (ex : 077) quitte à étendre ensuite les droits d'accès de certains fichiers.

Il ne faut pas quitter son terminal sans se déconnecter ou le verrouiller.

Il faut prêter attention aux dates de dernière connexion réussies et infructueuses qui sont affichées à chaque connexion.

Ne jamais autoriser l'accès, même en lecture, au fichier .profile.

Ne jamais mettre le . en première position du PATH, et contrôler ses chemins.

2. Contrôler les droits d'endossement

Les droits d'endossement (bits SUID et SGID) sont souvent une cause d'insécurité du système. En effet un utilisateur mal intentionné profitant de l'inattention ou de l'absence d'un collègue ou d'un administrateur n'étant pas déconnecté de sa console peut modifier les droits de certaines commandes à son avantage. L'exemple le plus courant est de recopier un shell en tant que programme peu utilisé (par exemple sx) et de lui donner les droits SUID. En lançant cette commande vous pouvez devenir root.

Obtenir le droit de lister tous les fichiers

```
# chmod u+s cat
```

Obtenir un shell root

```
# cp /bin/sh /bin/sx

# chmod u+s /bin/sx

...

$ sx

# ...
```

La commande suivante permet de rechercher tous les fichiers disposant des bits SUID et/ou SGID :

```
# find / -type f \( -perm -4000 -o -perm -2000 \)

# find / -type f \( -perm -4000 -o -perm -2000 \)

/bin/su
```

/bin/umount

/bin/eject

/bin/mount

/bin/ping

/bin/ping6

/sbin/unix2_chkpwd

/sbin/unix_chkpwd

/usr/bin/expiry

/usr/bin/write

/usr/bin/passwd

/usr/bin/newgrp

/usr/bin/gnotski

/usr/bin/mahjongg

/usr/bin/chfn

/usr/bin/yset

/usr/bin/wall

/usr/bin/crontab

/usr/bin/v4l-conf

/usr/bin/gnomine

/usr/bin/same-gnome

/usr/bin/gnotravex

/usr/bin/gnobots2

...

Dans la liste précédente il y a un intrus : **/usr/bin/yset** qui permet de modifier les réglages d'un serveur de son et qui n'a nul besoin d'avoir le droit SUID. Il y a un problème.

```
# ls -l /usr/bin/yset

-rwsr-sr-x 1 root root 604040 mai 19 21:28 /usr/bin/yset

# md5sum /usr/bin/yset

04ff72010ff1cf1c14d7706159cdf8bf  /usr/bin/yset

# ls -l /bin/bash

-rwxr-xr-x 1 root root 604040 sep 22  2007 /bin/bash

# md5sum /bin/bash

04ff72010ff1cf1c14d7706159cdf8bf  /bin/bash
```

Quelqu'un a recopié un shell avec un autre nom.

3. Vérifier les packages

Le chapitre Installation de Linux et de logiciels a abordé toute la gestion des packages logiciels. Parmi les diverses options, certaines permettent de contrôler la validité d'un package. La base RPM contient, outre le nom du fichier, son type (configuration, binaire, etc.) et dans certains cas (binaire) la somme de contrôle (checksum) MD5 du fichier.

Suivant l'exemple précédent, comment restaurer le fichier yset ? En trois étapes :

Trouvez le package d'origine :

```
# rpm -qf /usr/bin/yset

yiff-2.14.5-0.pm.1
```

Contrôlez l'état du package installé :

```
# rpm -V yiff

SM5....T    /usr/bin/yset
```

S : la taille n'est pas la bonne

M : les permissions ont été modifiées

5 : la somme de contrôle MD5 diffère

T : la date de modification n'est pas correcte.

Réinstallez le package d'origine, selon les modalités propres à votre distribution.

4. Politique de mot de passe

Les mots de passe sont à la base de l'authentification d'un utilisateur. Ils doivent être sûrs. C'est pourtant généralement une grosse lacune, tant au travail qu'à la maison, et même sur Internet :

mot de passe présent sur un post-it ;

emploi d'un gestionnaire de mot de passe automatique lui-même sans mot de passe ;

même mot de passe pour tous les sites Web et logiciels ;

mot de passe jamais changé ;

mot de passe et/ou compte communs à toute la famille/service ;

mot de passe pas assez complexe ;

etc.

Il ne sert à rien de sombrer dans la paranoïa. Vous devez trouver un compromis. Si vous demandez aux utilisateurs de modifier leur mot de passe trop souvent, ou s'il est trop difficile, ils auront tendance à le noter. S'il est trop simple et que vous attendez trop longtemps, il n'est plus sûr.

Les utilisateurs doivent choisir un bon mot de passe, en évitant la simplicité ou plutôt les évidences : noms des enfants, de sa femme, de lieux, dates de naissance et généralement tout ce qui tient à cœur et qui est connu du milieu professionnel ou personnel.

Un compromis peut être de modifier les règles de changement de mot de passe avec chage (ou passwd) de manière à placer une durée de validité des mots de passe de 40 jours. Les commandes de modification de la politique de gestion des mots de passe ont été présentées au chapitre Tâches administratives - Administration des utilisateurs.

```
# chage -l bean

Minimum:        7

Maximum:        40

Warning:        10

Inactive:       5

Last Change:            avr 10, 2008

Password Expires:       mai 20, 2008

Password Inactive:      mai 25, 2008
```

Account Expires: jan 01, 2010

Les modules PAM influent sur la politique de gestion des mots de passe, forçant éventuellement à en choisir un plus ou moins complexe. De manière surprenante, un mot de passe doit être assez facile à retenir par un utilisateur, ce qui n'implique pas forcément qu'il soit facile à pirater (par John the Riper par exemple). Il existe des mots de passe qui peuvent se retenir par des moyens mnémotechniques. Vous pouvez aussi générer des mots de passe automatiquement avec l'outil **pwgen**.

$ pwgen

uash6She lohJo7ae Ohphab3i ouRik9ie uM4va3im Neer7Eit eib3Hauy xo9Iuy5p

ahSiW0uf AhG6wail Yai6neeh phae4ioV deeL3aip Uz5ahzaa aiV5phee Aegaiy7x

ioPh1ahn Ong6Baib Eish4rip eik9Gie1 ien3Iepe xohduj7U aiP2keov So5ovaht

Voh9oxoe ahs2Meeg Ooch5xix Phe3yiuz eeCa5ohv aig9Ai3o Go4Ateeh Hee6thei

Rai6Daeh aid8ieNg Thah6ien daphaiG0 Iefai5oh Pheife6i Poora8ah Coh5Aida

ViC7ieth hohG5sei Aa9Jeilu eopoX8Si jooh3Eif dooPhai1 chohqu1G ieNgae3o

wiCeiSi3 aej6Piev eoTha1Fu ieR2yeeb Eireili6 saiGhie2 XohRoo1a cahb2Yah

Guungah0 ube3vo0D oshol3Op Pui6agh5 Ao7baeN1 foTek9Ei aeM3lala Ene2baol

geloV9ai Weeyu2ie Uvae2Vie dei0euL7 Xee9uaza ed8Eeghu eebiu2Ka zey0Liuh

be6Ailoi eiph8Ohb Yahpahr4 aij4dahG oQu2chae Fe5eeg9c Hoosh6oh Iip8eiwe

AuPie0um Ahxai9eo Dae5oquu Ie7Viek8 pa2aew8B fohham7A fah1Oogi ieH9vee8

saeC8sha Aejeey6i Eithoow1 yi9vei0L ohC7eegh IaTh4ohn ti6Fooxe Oiche7oh

Tah9uos7 Paej2Iec chuiD8ei aicoGh5l saiKeiw2 mae9mieY Ais9oanu Mah9xej3

Zi2nacai gaiM4thi sapa1Fah kie8oZo7 Po5uuho8 thae3Aim Ohjahgh9 Weike8ra

Cah4weiZ teoji0Oo vi0hei6O Zieha3ai Keip2bie bahR7bah ahSai0Ei afoh3Thi

eeNieTh8 Zei7eth8 uV5eichi kue1Eedi sueThe0V wohChe2u Ohl1zicu Loo1soo3

yahb9uSi EelieGh1 aeMiThi1 OoFoh8wu Ieyei5ka Roph7ape uem5quuK ahQu7eec

NahSha6A kooMou0y gu1chaiJ hae2ku0Z uC3oeNgo xuSha7qu Iucai0fu uK4icewe

eP7aetig ahYai0ee Eetahfu8 yeep0oPi Veimaij3 Oht0aiPh buTh9oob ood4nieC

sah7Ahj1 koozah0J Vieb9Bit eeP9nee1 ea1SohCe Afei4ohS eikahk0W rachoG4c

Ces mots de passe sont pseudo-aléatoires. Si vous parlez l'anglais (et que vous parlez geek/leet), ils s'écrivent comme ils se prononcent. Par exemple :

dooPhai1 : Do you fail ?

Vous pouvez demander à générer des mots de passe totalement aléatoires d'une longueur donnée, ici de 10 caractères :

$ pwgen -s -1 10

ER9BAgHsZH

5. Interdire les connexions

a. /bin/false

Certains comptes ne doivent pas être interactifs : les connexions depuis une console devraient leur être interdites. Ces comptes peuvent être dédiés à une application, à un service, à une connexion FTP, etc., mais la connexion devrait être refusée : pas de shell !

Dans la liste des shells autorisés, un attire l'attention :

$ cat /etc/shells

/bin/ash

/bin/bash

/bin/bash1

/bin/csh

/bin/false

/bin/ksh

/bin/sh

/bin/tcsh

/bin/true

/bin/zsh

/usr/bin/csh

/usr/bin/ksh

/usr/bin/passwd

/usr/bin/bash

/usr/bin/tcsh

/usr/bin/zsh

Le shell **/bin/false** n'en est pas vraiment un. Il interdit les connexions interactives. Vous avez déjà rencontré la commande **false** dans le chapitre Le shell et les commandes GNU. Elle retourne toujours faux. Dès que **login** tente d'exécuter le shell de connexion l'utilisateur est refoulé.

$ cat /etc/passwd|grep false

avahi:x:104:106:User for Avahi:/var/run/avahi-daemon:/bin/false

haldaemon:x:101:102:User for haldaemon:/var/run/hal:/bin/false

icecream:x:102:103:Icecream Daemon:/var/cache/icecream:/bin/false

mail:x:8:12:Mailer daemon:/var/spool/clientmqueue:/bin/false

messagebus:x:100:101:User for D-Bus:/var/run/dbus:/bin/false

ntp:x:74:104:NTP daemon:/var/lib/ntp:/bin/false

polkituser:x:103:105:PolicyKit:/var/run/PolicyKit:/bin/false

postfix:x:51:51:Postfix Daemon:/var/spool/postfix:/bin/false

sshd:x:71:65:SSH daemon:/var/lib/sshd:/bin/false

wwwrun:x:30:8:WWW daemon apache:/var/lib/wwwrun:/bin/false

vscan:x:65:110:Vscan account:/var/spool/amavis:/bin/false

b. /etc/nologin

Avant même de passer par un pseudo-shell de connexion, les modules PAM autorisent de nombreuses limitations. Parmi eux le module « **pam_nologin** » qui vous permet d'interdire la connexion des utilisateurs sauf root. Si un utilisateur tente de se connecter le contenu du fichier **/etc/nologin** est affiché. C'est utile en cas de maintenance : seul root peut alors se connecter.

$ pwd

/etc/pam.d

$ grep nologin *

login:auth requisite pam_nologin.so

ppp:auth required pam_nologin.so

sshd:auth requisite pam_nologin.so

Pensez aussi que vous pouvez interdire l'accès d'un compte donné via le module PAM « **pam_listfile** ».

c. /etc/securetty

Dans le même genre, le fichier **/etc/securetty** contient la liste des terminaux considérés comme sûrs. Pour le service donné, la connexion sera interdite si le terminal de la personne tentant de s'y connecter n'apparaît pas. Dans l'exemple suivant les logins (via la commande login) sont autorisés uniquement depuis les pseudo-terminaux locaux, c'est-à-dire seulement depuis les consoles directement accessibles sur l'ordinateur via les combinaisons [Alt][F1] à [Alt][F7].

$ grep securetty *

login:auth [user_unknown=ignore success=ok ignore=ignore

auth_err=die default=bad] pam_securetty.so

$ cat /etc/securetty

#

This file contains the device names of tty lines (one per line,

without leading /dev/) on which root is allowed to login.

#

tty1

tty2

tty3

tty4

tty5

tty6



Ne confondez pas securetty (Secure tty) avec security !

6. Tester les mots de passe

Les outils crack et « John the ripper » tentent de décrypter vos mots de passe, tant depuis un dictionnaire que par la force brute (les uns après les autres). Dans le pire des cas, ils les trouvent en quelques secondes ; dans le meilleur, en plusieurs jours, voire semaines. Dans ce cas, le mot de passe peut être considéré comme sûr.

"John the ripper" s'utilise très simplement. La commande est **john**. Pour tester l'intégralité de votre fichier **/etc/shadow** :

john /etc/shadow

Loaded 5 password hashes with 5 different salts (OpenBSD Blowfish

[32/64])

Dans le mode par défaut, john :

tente une détection simple via des combinaisons courantes liées au compte,

passe au mode dictionnaire avec application de règles,

puis tente une recherche incrémentale.

Une recherche peut prendre de quelques secondes à quelques semaines !

Pour tester un seul utilisateur :

john -user:seb /etc/shadow

Pour tester les utilisateurs avec un dictionnaire :

john -users:seb -wordlist:/var/lib/john/wordlists/all /etc/shadow

La même chose mais en testant plusieurs règles par mot (inversion, majuscules, minuscules, etc.) :

john -users:seb -wordlist:/var/lib/john/wordlists/all -rules

/etc/shadow

Pour reprendre une recherche interrompue ([Ctrl][C]) :

john -restore

John place ses résultats dans le répertoire ~/.john, généralement chez root qui seul, en principe, devrait pouvoir lancer cet outil :

ls -l .john

total 4

-rw------- 1 root root 70 mai 23 21:58 john.pot

-rw------- 1 root root 124 mai 23 21:54 john.rec

Le fichier **john.pot** contient les résultats trouvés par John. Ici le fichier n’est pas vide. C’est problématique : john a trouvé un mot de passe. Le fichier **john.rec** contient l’état actuel de la recherche, utilisé en interne et en cas de reprise après interruption.

Sur une machine basée sur un Intel Core 2 duo 64 bits à 3.4 Ghz, le mot de passe de seb (celui de l’auteur) a été cracké en 4 minutes et 22 secondes. Il était (volontairement cette fois) basé sur un mot du dictionnaire.

Si vous appuyez sur une touche durant la recherche, john affiche son état.

Soit un compte henri dont le propriétaire a pour mot de passe le même mot que son nom de compte, à savoir « henri » :

john -users:henri /etc/shadow

Created directory: /root/.john

Loaded 1 password hash (OpenBSD Blowfish [32/64])

henri (henri)

guesses: 1 time: 0:00:00:00 100% (1) c/s: 4.34 trying: henri

Voici de quoi faire grandement réfléchir ceux qui pensent que personne ne viendra les déranger, et les encourager à modifier leur mot de passe avec des règles précises (majuscules, minuscules, chiffres, etc.).

7. Rechercher des rootkits

a. Principe du rootkit

Une fois qu’un pirate quelconque aura réussi, via une faille ou un mot de passe trop simple, à pénétrer votre machine, il cherchera probablement à s’aménager une porte d’entrée plus importante, ou plutôt une porte de derrière, une **backdoor**, afin de pouvoir revenir se servir de votre machine à des fins douteuses ou y puiser ou stocker des données (certains PCs se sont retrouvés comme cela à contenir des milliers de fichiers mp3 ou divx et à servir de serveur pour des personnes peu scrupuleuses).

L’idéal pour le pirate est de pouvoir s’accaparer les droits de root. C’est son but : devenir intégralement le maître de votre machine. Pour cela il n’a pas forcément besoin de « taper » directement sur ce compte. Sous Linux, il est courant (et voir conseillé) de se connecter en

simple utilisateur puis de passer root le temps d’effectuer les manipulations nécessaires. Or pour passer root, vous utilisez généralement **su**.

Si, via le service ftp mal configuré, ou via ssh (et scp ensuite), le pirate tente de se connecter à un compte dont le mot de passe est évident (exemple du compte henri), qu’il y place un script appelé su de son cru et qu’il modifie le PATH par défaut pour y placer son chemin en premier, alors c’est gagné :

$ pwd

/home/seb

$ cat su

#!/bin/bash

echo -e "Mot de passe :\c"

read -s password

echo "$@ $password" > /tmp/fic

echo

echo "su: Mot de passe incorrect."

/bin/su $@

$ chmod +x su

$ export PATH=$HOME:$PATH

$ su - root

Mot de passe : ===> FAUX SU

su: Mot de passe incorrect. ===> FAUX SU

Mot de passe : ===> VRAI SU

Il n’y a plus qu’à afficher le contenu du fichier pour obtenir le mot de passe :

$ cat /tmp/fic

- root azerty

La méthode, simpliste mais efficace, est loin d’être imparable : su ne demande le mot de passe qu’une seule fois, et sauf faute d’inattention le subterfuge est ici volontairement visible. Le fait de placer des scripts, de modifier l’environnement, de remplacer un fichier par un autre de

manière à obtenir un accès privilégié à une machine s'appelle installer un **rootkit**. Une fois celui-ci en place, il garantit, tant qu'il n'a pas été détecté, un accès root à la machine.

b. chkrootkit

L'outil **chkrootkit** est un outil simple permettant de rechercher la présence des rootkits les plus connus et les plus courants. Il est efficace seulement s'il est mis à jour régulièrement, lancé régulièrement, et ne se substitue pas aux contrôles déjà cités précédemment.

```
# chkrootkit

ROOTDIR is `/'

Checking `amd'... not found

Checking `basename'... not infected

Checking `biff'... not found

Checking `chfn'... not infected

Checking `chsh'... not infected

Checking `cron'... not infected

Checking `crontab'... not infected

...

Searching for sniffer's logs, it may take a while... nothing found

Searching for HiDrootkit's default dir... nothing found

Searching for t0rn's default files and dirs... nothing found

Searching for t0rn's v8 defaults... nothing found

Searching for Lion Worm default files and dirs... nothing found

Searching for RSHA's default files and dir... nothing found

Searching for RH-Sharpe's default files... nothing found

Searching for Ambient's rootkit (ark) default files and dirs...

nothing found

Searching for suspicious files and dirs, it may take a while...
```

...

Searching for LPD Worm files and dirs... nothing found

Searching for Ramen Worm files and dirs... nothing found

Searching for Maniac files and dirs... nothing found

Searching for RK17 files and dirs... nothing found

Searching for Ducoci rootkit... nothing found

Searching for Adore Worm... nothing found

Searching for ShitC Worm... nothing found

Searching for Omega Worm... nothing found

Searching for Sadmind/IIS Worm... nothing found

...

Searching for ENYELKM rootkit default files... nothing found

Searching for anomalies in shell history files... Warning: `' is

linked to another file

Checking `asp'... not infected

Checking `bindshell'... not infected

Checking `lkm'... chkproc: nothing detected

Checking `rexedcs'... not found

Checking `sniffer'... eth0: not promisc and no PF_PACKET sockets

vmnet8: not promisc and no PF_PACKET sockets

vmnet1: not promisc and no PF_PACKET sockets

Checking `w55808'... not infected

...

8. Les virus

Les premiers virus sont apparus sous Unix. Si le système est généralement sécurisé, et que les virus sur les plates-formes Unix et Linux (dont MacOS X) sont presque inexistants, dans le cas plus ou moins probable où un virus serait présent et risque de compromettre la sécurité de votre machine, celle des autres ou de tout un réseau, il est de votre devoir de l'éradiquer.

Si votre machine sert de serveur, notamment de courrier électronique ou de fichiers sur un réseau contenant des machines sous Windows fortement exposées, vous ne devez pas servir de vecteur indirect de propagation. Vous devez éliminer la menace.

Il existe plusieurs antivirus sous Linux, certains commerciaux gratuits ou non, certains libres. L'antivirus Clam (ClamAV) est libre et gratuit. Il est disponible à l'adresse http://www.clamav.net/. Ses bases de signatures sont mises à jour quotidiennement.

Freshclam permet de mettre à jour les bases placées dans **/var/lib/clamav** :

```
# pwd

/var/lib/clamav

# freshclam

ClamAV update process started at Sun May 25 16:29:37 2008

connect_error: getsockopt(SO_ERROR): fd=4 error=113: No route to host

Can't connect to port 80 of host database.clamav.net (IP: 194.116.142.73)

Trying host database.clamav.net (213.251.187.177)...

Downloading main.cvd [100%]

main.cvd updated (version: 46, sigs: 231834, f-level: 26, builder: sven)

Downloading daily.cvd [100%]

daily.cvd updated (version: 7233, sigs: 69750, f-level: 26, builder: ccordes)

Database updated (301584 signatures) from database.clamav.net (IP: 213.251.187.177)

# ll

total 14628

-rw-r--r-- 1 vscan vscan  1894380 mai 25 16:29 daily.cvd

-rw-r--r-- 1 vscan vscan 13050207 mai 25 16:29 main.cvd

-rw------- 1 vscan vscan       52 mai 25 16:29 mirrors.dat
```

Scanclam permet de rechercher les éventuels virus :

clamscan -v -r /usr/local/bin

Scanning /usr/local/bin/avidemux2_qt4

/usr/local/bin/avidemux2_qt4: OK

Scanning /usr/local/bin/avidemux2_gtk

/usr/local/bin/avidemux2_gtk: OK

Scanning /usr/local/bin/avidemux2_cli

/usr/local/bin/avidemux2_cli: OK

Scanning /usr/local/bin/i18n/qt_it.qm

/usr/local/bin/i18n/qt_it.qm: OK

Scanning /usr/local/bin/i18n/avidemux_it.qm

/usr/local/bin/i18n/avidemux_it.qm: OK

----------- SCAN SUMMARY -----------

Known viruses: 300812

Engine version: 0.93

Scanned directories: 2

Scanned files: 5

Infected files: 0

Data scanned: 45.64 MB

Time: 3.362 sec (0 m 3 s)

Voici le même test effectué avec un faux virus sous trois formes : binaire, compressé gzip et compressé bzip2. Si un virus est détecté le fichier correspondant est déplacé dans **/home/seb/VIRUS** :

$ clamscan -v -r --move=/home/seb/VIRUS /home/seb/bin

Scanning /home/seb/bin/eicarcom2.zip

/home/seb/bin/eicarcom2.zip: Eicar-Test-Signature FOUND

/home/seb/bin/eicarcom2.zip: moved to '/home/seb/VIRUS//eicarcom2.zip'

Scanning /home/seb/bin/eicar_com.zip

/home/seb/bin/eicar_com.zip: Eicar-Test-Signature FOUND

/home/seb/bin/eicar_com.zip: moved to '/home/seb/VIRUS//eicar_com.zip'

Scanning /home/seb/bin/eicar.com

/home/seb/bin/eicar.com: Eicar-Test-Signature FOUND

/home/seb/bin/eicar.com: moved to '/home/seb/VIRUS//eicar.com'

----------- SCAN SUMMARY -----------

Known viruses: 300812

Engine version: 0.93

Scanned directories: 1

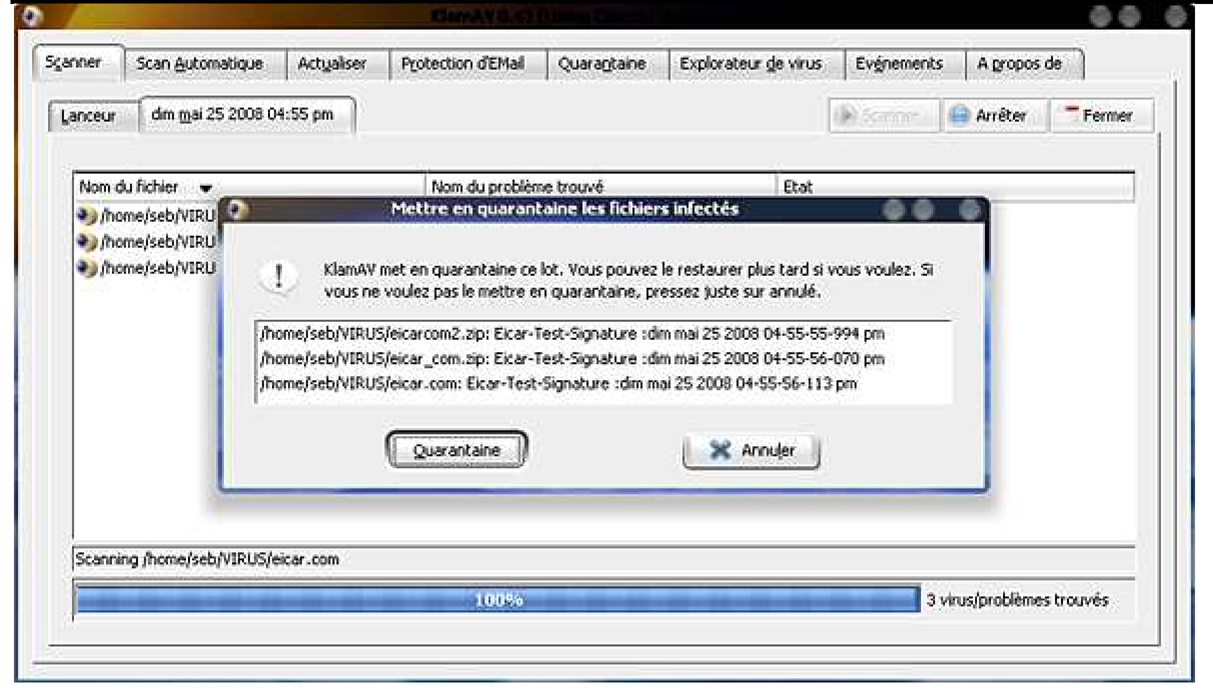
Scanned files: 3

Infected files: 3

Data scanned: 0.00 MB

Time: 1.216 sec (0 m 1 s)

Clamav peut être lancé en tant que service. Dans ce cas, il peut accepter des demandes de recherche de virus dans des arborescences précisées ce qui permet à des outils comme klamav de pouvoir exister : un frontend à clamav sous KDE.



Klamav a bien détecté le virus de test.

9. Les limites de l'utilisateur

Le champ d'action des PAM est plus vaste que la simple connexion puisqu'il gère aussi l'environnement de l'utilisateur. Avant même de voir le module concerné, la commande **ulimit** permet d'agir sur l'environnement du shell et des processus qu'il contrôle. Le paramètre **-a** affiche toutes les options contrôlées par ulimit :

$ ulimit -a

core file size (blocks, -c) 0

data seg size (kbytes, -d) unlimited

scheduling priority (-e) 0

file size (blocks, -f) unlimited

pending signals (-i) 16380

max locked memory (kbytes, -l) 32

max memory size (kbytes, -m) 1753125

open files (-n) 1024

pipe size (512 bytes, -p) 8

POSIX message queues (bytes, -q) 819200

real-time priority (-r) 0

stack size (kbytes, -s) 8192

cpu time (seconds, -t) unlimited

max user processes (-u) 16380

virtual memory (kbytes, -v) 3333600

file locks (-x) unlimited

Les quelques lignes en gras méritent votre attention.

max memory size : la taille mémoire max que l'utilisateur peut occuper ;

open files : le nombre maximum de descripteurs de fichiers, donc le nombre maximum de fichiers pouvant être ouverts ;

max user processes : le nombre maximum de processus qu'un utilisateur peut lancer.

Ces valeurs peuvent être changées selon certaines limites imposées par l'administrateur. Il existe des limites soft (douces ou basses) qui sont les valeurs par défaut retournées par ulimit, et des limites hard (dures, hautes) qui ne peuvent être dépassées.

Pour passer le nombre maximum de fichiers ouverts à 2048 :

$ ulimit -n 2048

$ ulimit -n

2048

L'administrateur root peut contrôler les valeurs par défaut grâce au fichier **/etc/security/limits.conf**.

$ grep seb /etc/security/limits.conf

seb hard nproc 32768

seb soft nofile 1024

seb hard nofile 4096

Dans cet exemple l'utilisateur seb est limité à un maximum de 32768 processus, peut ouvrir par défaut 1024 fichiers mais peut monter, via une action ulimit de sa part, à 4096.

10. Audit plus complet

Pour effectuer l'audit d'un système, vous pouvez utiliser, en plus de l'accès aux traces et aux historiques, et des commandes déjà répertoriées, des produits libres ou gratuits comme tripwire qui vérifie l'intégrité du système, COPS qui surveille la sécurité du système, Crack qui détecte les mauvais mots de passe...

11. Les bulletins de sécurité

a. CERT : Computer Emergency Response Team

Historique

C'est à un étudiant de l'université de Cornell que vous devez le premier virus capable de se répliquer sur le réseau Internet (qui s'appelait Arpanet à l'époque). Développé et lâché fin 1998 sans intention de nuire, ce programme se propageait et se répliquait tout seul en exploitant des failles de sécurité de Unix et de ses services. Ce programme a rapidement saturé le réseau Internet et les machines qu'il avait atteintes, paralysant le réseau Internet, alors composé de 60000 ordinateurs, pour plusieurs jours. Seulement 4% des machines avaient été infectées.

L'éradication de ce virus a nécessité du temps et beaucoup de moyens, fournis par le MIT, Berkeley, etc. Il a fallu étudier son fonctionnement pour comprendre comment il avait procédé et partant de là les trous de sécurité des systèmes et des serveurs ont pu être corrigés. Des patches correctifs furent diffusés. Le DARPA, initiateur du projet Arpanet (puis Internet) mit alors en place une nouvelle structure appelée le CERT (CERT/CC, **CERT Coordination Center**) chargée d'analyser les menaces futures et la vulnérabilité des systèmes.

Internet représente des dizaines de millions de machines interconnectées, avec des systèmes d'exploitation et des services différents. Le CERT (http://www.cert.org/) dont le siège est aujourd'hui à l'université de Canergie Mellon continue d'étudier les éventuelles vulnérabilités sur Internet, et ce même à long terme, afin de tenter d'obtenir la meilleure sécurité possible.

Rôle du CERT

Les missions du CERT sont les suivantes :

centralisation et analyse des demandes d'assistance suite aux incidents de sécurité (attaques) sur les réseaux et les systèmes d'information : réception des demandes, analyse des symptômes et éventuelle corrélation des incidents ;

traitement des alertes et réaction aux attaques informatiques : analyse technique, échange d'informations avec d'autres CERTs, contribution à des études techniques spécifiques ;

établissement et maintenance d'une base de donnée des vulnérabilités ;

prévention par diffusion d'informations sur les précautions à prendre pour minimiser les risques d'incident ou au pire leurs conséquences ;

coordination éventuelle avec les autres entités : centres de compétence réseaux, opérateurs et fournisseurs d’accès à Internet, CERTs nationaux et internationaux.

Bulletins du CERT

Il existe plusieurs comités CERT : par pays, pour l’industrie, etc. Les bulletins sont émis indépendamment mais peuvent être repris d’un CERT à l’autre. Voici deux sites où trouver des alertes :

http://www.certa.ssi.gouv.fr/site/index.html

Et surtout, http://www.us-cert.gov/

US-CERT Technical Cyber Security Alert TA08-137A -- Debian/Ubuntu OpenSSL Random Number Generator Vulnerability - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils Aide
http://www.us-cert.gov/cas/techalerts/TA08-137A.html
openSUSE ServiceCenter Portail AssetCenter Actualités agendax.lamulati... WEB XPLS Site de metrologi... Yahoo! Actualités...

Friday, May 23, 2008 Home | FAQ | Contact | Privacy & Use

US-CERT
UNITED STATES COMPUTER EMERGENCY READINESS TEAM

Security Publications | Alerts and Tips | Related Resources | About Us | Search US-CERT: | customize

National Cyber Alert System
Technical Cyber Security Alert TA08-137A
Archive

Debian/Ubuntu OpenSSL Random Number Generator Vulnerability

Original release date: May 16, 2008
Last revised: --
Source: US-CERT

Systems Affected

- Debian, Ubuntu, and Debian-based distributions

Overview

A vulnerability in the OpenSSL package included with the Debian GNU/Linux operating system and its derivatives may cause weak cryptographic keys to be generated. Any package that uses the affected version of SSL could be vulnerable.

I. Description

A vulnerability exists in the random number generator used by the OpenSSL package included with the Debian GNU/Linux, Ubuntu, and other Debian-based operating systems. This vulnerability causes the generated numbers to be predictable.

The result of this error is that certain encryption keys are much more common than they should be. This vulnerability affects cryptographic applications that use keys generated by the flawed versions of the OpenSSL package. Affected keys include, but may not be limited to, SSH keys, OpenVPN keys, DNSSEC keys, and key material for use in X.509 certificates and session keys used in SSL/TLS connections. Any of these keys generated using the affected systems on or after 2006-09-17 may be vulnerable. Keys generated with GnuPG, GNUTLS, ccrypt, or other encryption utilities that do not use OpenSSL are not vulnerable because these applications use their own random number generators.

II. Impact

Téléchargements firmware-8.10... Retirer
Terminé

Une alerte de sécurité sur US-CERT

Un exemple d’alerte est une faille de sécurité de la bibliothèque OpenSSL sous Debian « **Debian/Ubuntu OpenSSL Random Number Generator Vulnerability** », Référence TA08-137A. Les clés aléatoires générées depuis la version OpenSSL de Debian et Ubuntu ne le sont pas vraiment, ce qui limite fortement la sécurité de ces clés et qui pose un gros problème puisque la mise à jour de la bibliothèque ne suffit pas ; il faut aussi regénérer les clés existantes...

b. Bugtraq

Bugtraq est une liste de diffusion électronique (e-mailing) qui existe depuis 1993 et qui regroupe des discussions sur les vulnérabilités, la sécurité, les annonces, les moyens d'exploiter les failles et comment les corriger. La liste a été créée à l'origine à cause de deux problèmes :

les nombreux échecs du CERT à prévenir les problèmes ;

le laisser-aller de nombreux éditeurs et constructeurs qui ne fournissaient pas de mises à jours de sécurité malgré les failles découvertes.

Bugtraq appartenait à SecurityFocus, lui-même appartenant aujourd'hui à l'éditeur de suites de sécurité Symantec.

Vous pouvez vous inscrire à la mailing-list depuis le site http://www.securityfocus.com/archive.

Dans cette même page vous avez accès (premier cadre en haut à gauche) à l'historique (archives) de la liste.

Liste des bulletins de sécurité sur SecurityFocus

c. Les bulletins des distributions

Chaque éditeur des plus grandes distributions fournit aussi des bulletins de sécurité. Ils reprennent eux-mêmes les alertes de sécurité d'autres organismes comme le CERT ou des listes Bugtraq, mais comme chaque éditeur est responsable des packages qu'il diffuse, il doit diffuser lui-même un correctif d'autant plus que certains produits sont allégrement patchés par l'éditeur et diffèrent fortement de l'original. D'où l'émission d'une alerte pour prévenir ses clients et utilisateurs. Voici les endroits où vous pouvez obtenir des informations de sécurité pour les principales distributions :

Debian : http://www.debian.org/security/

openSUSE : http://lists.opensuse.org/opensuse-security-announce/

Fedora : https://www.redhat.com/archives/fedora-security-list/

Ubuntu : http://www.ubuntu.com/usn

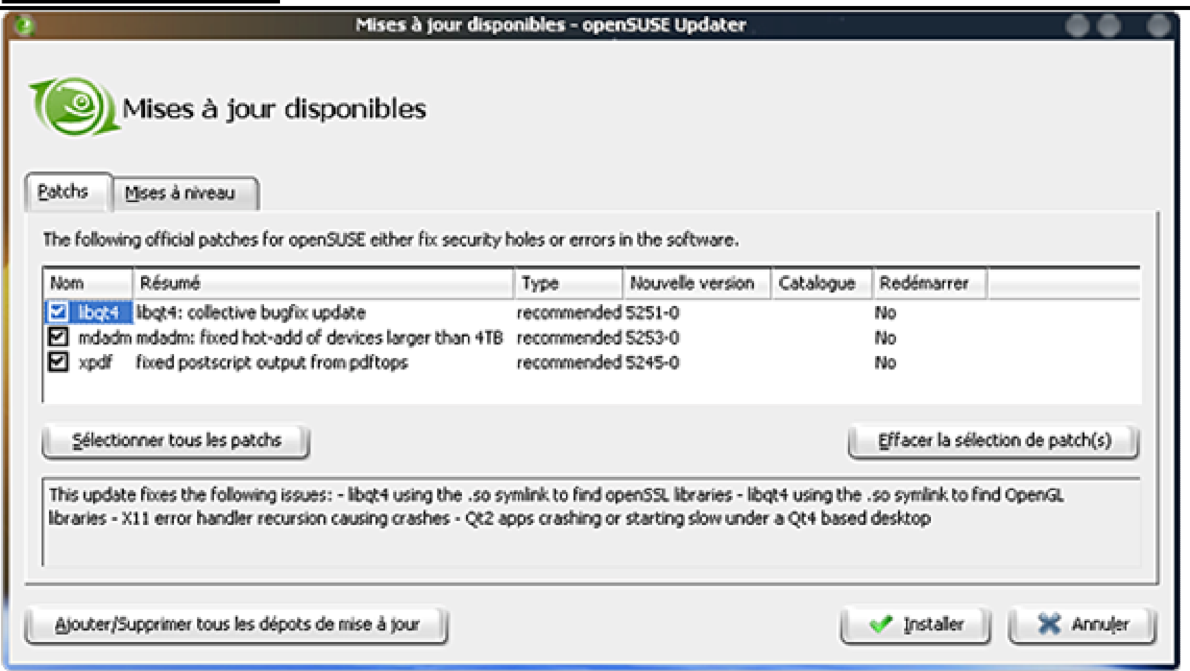
Mandriva : http://www.mandriva.com/security/advisories

Red Hat Enterprise : http://www.redhat.com/security/updates/errata/

d. Les correctifs

Il ne suffit pas de fournir une alerte lorsqu'un trou de sécurité est détecté, il faut aussi le réparer. Les éditeurs fournissent pour cela soit des packages corrigés (le lien est souvent dans le bulletin d'alerte), soit des patchs correctifs. Généralement chaque distribution est fournie avec un composant logiciel permettant de récupérer ces patches, mais aussi de vous informer de leur disponibilité.

Sur openSUSE, le produit openSUSE Updater se place dans la barre des tâches (ou plutôt la boîte à miniatures de KDE ou son équivalent sur d'autres environnements) et vous informe de la présence de mises à jour. Vert : pas de mises à jours ou mises à jour mineures ; orange : mises à jour recommandées (pas de sécurité) ; rouge : mises à jour critiques.



Les mises à jour de bug et de sécurité par openSUSE Updater.

Chaque distribution est différente de ce point de vue : vous devez vous renseigner.

TP : La sécurité - Contrôle des fichiers & Sécurité des utilisateurs

114.2 Sécurité de la machine locale (3)

Sécurité des services et du réseau

1. Vérifier les ports ouverts

a. Les sockets

Les connexions réseau entre deux machines s’effectuent par des **sockets**. Une socket est une connexion entre deux points via un réseau. Une machine dispose d’une adresse IP et de ports (virtuels) de connexion numérotés, auxquels sont rattachés des services (voir pour cela le chapitre Le réseau). Un client établit une connexion depuis un port de sa machine (port > 1024, généralement choisi aléatoirement parmi les ports libres) vers un port donné d’une autre machine, par exemple un serveur Web sur le port 80. La communication établie entre les deux passe par une socket.

Le système peut être configuré pour accepter ou rejeter des connexions depuis ou vers certains ports locaux ou distants, idem pour les adresses IP. C’est le rôle du firewall (mur de feu) comme **Netfilter**.

Sur une installation Linux de base, sauf si l’option était présente lors de l’installation, le firewall n’est pas toujours activé par défaut. Les ports ne sont pas filtrés et toute machine

extérieure peut tenter d'établir une connexion sur un port de votre machine : ce qui s'appelle ouvrir une socket.

Cela ne signifie pas forcément qu'il y a un trou de sécurité : si aucun service n'est à l'écoute, la connexion est impossible. C'est cependant rarement le cas. De nombreux services sont présents et démarrés par défaut. Si certains souffrent de vulnérabilités, ou sont configurés de manière trop laxiste, il existe un risque réel d'intrusion.

b. Informations depuis netstat

Le chapitre Le réseau vous a présenté l'outil **netstat** qui permet d'obtenir des informations et des statistiques réseau sur une machine locale. Notamment vous pouvez vérifier quels sont les ports à l'écoute sur votre machine, qui a établi une connexion, et quels sont les processus (services) locaux à l'écoute :

```
# netstat -a -A inet -p

Connexions Internet actives (serveurs et établies)

Proto Recv-Q Send-Q Adresse locale          Adresse distante

Etat       PID/Program name

tcp        0      0 *:5800                  *:*

LISTEN     31232/kded [kdeinit

tcp        0      0 *:5801                  *:*

LISTEN     23224/xinetd

tcp        0      0 *:vnc-server            *:*

LISTEN     31232/kded [kdeinit

tcp        0      0 *:5901                  *:*

LISTEN     23224/xinetd

tcp        0      0 *:sunrpc                *:*

LISTEN     3076/portmap

tcp        0      0 *:ndmp                  *:*

LISTEN     26746/sshd

tcp        0      0 *:6000                  *:*
```

```
LISTEN      31088/Xorg

tcp        0      0 *:7634                *:*

LISTEN      3006/hddtemp

tcp        0      0 *:ftp                 *:*

LISTEN      26772/vsftpd

tcp        0      0 localhost:ipp         *:*

LISTEN      3092/cupsd

tcp        0      0 localhost:smtp        *:*

LISTEN      3159/master

tcp        0      0 slyserver:38144       eeepc:ssh

ESTABLISHED 25622/ssh

tcp        0      0 slyserver:45046       eeepc:ms-wbt-server

ESTABLISHED 7803/rdesktop

tcp        0      0 slyserver:ftp         eeepc:49502

FIN_WAIT2   -

udp        0      0 *:xdmcp               *:*                     27850/kdm

udp        0      0 *:59742               *:*                     2984/avahi-daemon:

udp        0      0 *:mdns                *:*                     2984/avahi-daemon:

udp        0      0 *:sunrpc              *:*                     3076/portmap

udp        0      0 *:ipp                 *:*                     3092/cupsd
```

Les lignes en gras montrent trois connexions établies ou terminées entre les machines slyserver et eeepc.

l'eeepc était connecté sur le port ftp de slyserver,

slyserver est connecté sur le port ssh de l'eeepc,

slyserver est connecté sur le port ms-wbt-server de l'eeepc (protocole terminal server).

c. L'outil nmap

Il existe une armada d'outils de sécurité, de vérification, de tests, etc. L'outil **nmap** en fait partie. Il se définit comme un outil d'exploration réseau et d'audit de sécurité. Il permet de tester les connexions réseaux d'une machine donnée et de retourner un grand nombre d'informations. Notamment, en analysant les trames, il arrive souvent à déterminer le type et la version du système d'exploitation distant.

Le mode d'emploi de nmap fait plus de 2000 lignes, il est donc impossible d'étudier l'ensemble du produit. Mais voici quelques possibilités.

Examinez les ports à l'écoute sur la machine de test ayant servi à la rédaction de ce livre. Plusieurs ports sont ouvert (des services sont à l'écoute). Certains d'entre eux peuvent présenter des risques : telnet, netbios-ssn et microsoft-ds (partages windows), ftp, vnc, etc.

```
# nmap localhost

Starting Nmap 4.20 ( http://insecure.org ) at 2008-05-22 14:41 CEST

Interesting ports on localhost (127.0.0.1):

Not shown: 1684 closed ports

PORT     STATE SERVICE

21/tcp   open  ftp

23/tcp   open  telnet

25/tcp   open  smtp

111/tcp  open  rpcbind

139/tcp  open  netbios-ssn

445/tcp  open  microsoft-ds

631/tcp  open  ipp

5800/tcp open  vnc-http

5801/tcp open  vnc-http-1

5900/tcp open  vnc

5901/tcp open  vnc-1
```

6000/tcp open X11

10000/tcp open snet-sensor-mgmt

Nmap finished: 1 IP address (1 host up) scanned in 0.170 seconds

En quelques commandes il est possible de désactiver les services non nécessaires :

chkconfig telnet off

service vsftpd stop

service smb stop

service nmb stop

service xinetd restart

nmap localhost

Starting Nmap 4.20 (http://insecure.org) at 2008-05-22 14:48 CEST

Interesting ports on localhost (127.0.0.1):

Not shown: 1688 closed ports

PORT   STATE SERVICE

25/tcp   open smtp

111/tcp   open rpcbind

631/tcp   open ipp

5800/tcp open vnc-http

5801/tcp open vnc-http-1

5900/tcp open vnc

5901/tcp open vnc-1

6000/tcp open X11

10000/tcp open snet-sensor-mgmt

Dès qu’un service réseau est arrêté, le port n’est plus accessible.

Le paramètre **-A** permet de détecter en plus le système d'exploitation distant et sa version. Pour cela un ou plusieurs ports doivent être ouverts. Mieux : nmap interroge chaque service associé aux ports trouvés quand c'est possible pour récupérer des informations. Notez en gras les valeurs remarquables :

nmap -A machine

Starting Nmap 4.20 (http://insecure.org) at 2008-05-22 19:54 CEST

Interesting ports on machine.mondomaine.com (192.168.1.25):

Not shown: 1676 closed ports

PORT STATE SERVICE VERSION

9/tcp open discard

13/tcp open daytime

21/tcp open ftp **vsftpd 2.0.5**

22/tcp open ssh **OpenSSH 4.3p2 Debian 9 (protocol 2.0)**

25/tcp open smtp **Postfix** smtpd

37/tcp open time (32 bits)

53/tcp open domain

80/tcp open http **Apache httpd 2.2.3 ((Debian) PHP/5.2.0-8+etch10)**

111/tcp open rpcbind 2 (rpc #100000)

113/tcp open ident OpenBSD identd

139/tcp open netbios-ssn **Samba smbd 3.X (workgroup: SLYNET)**

199/tcp open smux Linux SNMP multiplexer

445/tcp open netbios-ssn Samba smbd 3.X (workgroup: SLYNET)

606/tcp open mountd 1-2 (rpc #100005)

631/tcp open ipp **CUPS 1.2**

901/tcp open http Samba SWAT administration server

1389/tcp open ldap   **OpenLDAP 2.2.X**

2049/tcp open nfs   2 (rpc #100003)

3128/tcp open squid-http

7937/tcp open nsrexec   1 (rpc #390113)

7938/tcp open rpcbind   2 (rpc #100000)

Device type: general purpose

Running: Linux 2.6.X

OS details: Linux 2.6.14 - 2.6.17

Uptime: 112.678 days (since Wed Jan 30 21:39:53 2008)

Network Distance: 2 hops

Service Info: Host: machine.mondomaine.org; OSs: Unix, Linux, OpenBSD

OS and Service detection performed. Please report any incorrect results

at http://insecure.org/nmap/submit/ .

Nmap finished: 1 IP address (1 host up) scanned in 115.360 seconds

Ainsi, une machine dispose au final de bien peu de secrets si elle est mal protégée. Le serveur est sous Linux Debian Etch disposant d'un noyau compris entre 2.6.14 et 2.6.17 (une erreur de détection ici, le noyau est un 2.6.18). Les services :

OpenSSH 4.3p2

Apache 2.2.3 et PHP 5.2.0

Postfix

Samba 3.x

Cups 1.2

OpenLDAP 2.2.X

Vsftpd 2.0.5

Il suffit d'aller consulter les bulletins de sécurité pour voir si l'une de ces versions est connue pour avoir des problèmes de sécurité, et si c'est le cas le serveur présente un risque.

2. Supprimer les services inutiles

a. Généralités

Si vous vous faites « hacker » votre système, c'est que la personne mal intentionnée a trouvé le moyen de rentrer. Contrairement à l'idée répandue, l'installation d'un firewall ne résout pas tous les problèmes, d'autant plus que sur un poste de travail la tendance est d'ouvrir plusieurs ports réseaux vers Internet (ou plutôt depuis Internet) : ftp, http, p2p (réseaux eDonkey, c'est-à-dire eMule, Bittorrent, etc.), ssh et ainsi de suite. Or il suffit qu'un seul des services associés présente un risque pour que votre machine soit attaquée avec les problèmes qui en découlent.

Même si le service en tant que tel n'a pas de trou connu, le paramétrage que vous avez appliqué peut être trop simple ou laxiste. Il ne serait pas très malin de laisser votre serveur ssh accepter les connexions depuis l'extérieur si votre mot de passe ou celui de root est toto, password ou quelque chose de ressemblant. Lors d'une attaque l'auteur de ce livre a eu l'occasion de constater que l'attaquant tentait en boucle de se connecter, via ssh, en utilisant une série de logins/mots de passe prédéfinis parmi certaines combinaisons classiques préconfigurées par défaut dans certains programmes. Un cas simple est une installation de MySQL par défaut où le compte d'administration n'a pas de mot de passe.

Donc, pensez à désactiver tous les services dont vous n'avez pas besoin. S'il s'avère que certains vous sont nécessaires à certains moments et pas à d'autres, n'hésitez pas à les démarrer seulement à ce moment, et à les stopper ensuite. De même, sur votre firewall (netfilter ou autre) ne laissez ouverts que les ports strictement nécessaires.

b. Services standalone

Les services standalone, c'est-à-dire lancés de manière indépendantes, sont contrôlés via la commande **service** ou **/etc/init.d/service**. Pour contrôler les arrêts et relances de ces services de manière pérenne, utilisez les commandes **chkconfig** (distributions RPM) ou **rcudpate.d** (Debian).

c. Services xinetd

Les services contrôlés par xinetd peuvent être activés et désactivés par la ligne « **disable** » de leur fichier de configuration.

```
$ pwd

/etc/xinetd.d

$ grep disable *

chargen:        disable        = yes

chargen-udp:    disable        = yes
```

cups-lpd: disable = yes

...

vmware-authd: disable = no

vnc: disable = yes

...

Pour prendre en compte les changements, forcez xinetd à recharger sa configuration.

/etc/init.d/xinetd reload

3. Les tcp_wrappers

Les **enveloppeurs TCP** ou tout simplement **tcp_wrappers** permettent la vérification des accès à un service réseau donné (service, xinetd, portmapper). Chaque programme utilisant les tcp_wrappers est compilé avec la bibliothèque **libwrap** de manière statique (la commande ldd ne permet pas de voir la bibliothèque).

Pour savoir si un service réseau est compilé avec libwrap, on saisit la commande suivante :

strings -f <binaire> | grep hosts_access

Voici un exemple avec xinetd qui utilise les tcp_wrappers :

strings -f /usr/sbin/xinetd |grep hosts_access

/usr/sbin/xinetd: hosts_access

Si aucune ligne n'est retournée, le programme n'utilise pas les tcp_wrappers.

Parmi les services utilisant les tcp_wrappers, on trouve :

sendmail (y compris postfix),

sshd (ssh),

xinetd (et donc indirectement tous les services associés),

vsftpd (ftp),

portmap (et donc nis, nfs),

in.telnetd (telnet) ainsi que la plupart des services supportés par xinetd,

dovecot (imap, pop).

La vérification d’accès à un service enveloppé TCP a lieu en trois étapes :

l’accès est-il explicitement autorisé ?

sinon, l’accès est-il explicitement interdit ?

sinon, par défaut, l’accès est autorisé.

Les fichiers de configuration sont **/etc/hosts.allow** et **/etc/hosts.deny**. La syntaxe est commune :

daemon_list : client_list [:options]

daemon_list : liste des **exécutables (PAS DES SERVICES)** séparés par des virgules. Vous pouvez mettre **ALL** pour spécifier tous les services. Si vous disposez de plusieurs interfaces réseau on peut utiliser la syntaxe avec @ : service@ip.

in.telnetd: ...

sshd, portmap: ...

sshd@192.168.1.7 : ...

client_list : clients autorisés ou interdits pour ce service. On peut spécifier l’adresse IP, le nom, le masque de réseau, le nom du réseau, etc.

... : 192.168.1.7, 192.168.1.8

... : 192.168.1.

... : poste1, poste2

... : 192.168.1.0/255.255.255.0

... : .mondomaine.org

La liste des clients admet une syntaxe avancée :

ALL : correspondance systématique.

LOCAL : tous les hôtes dont le nom ne contient pas de point (poste1, poste2, etc.).

UNKNOWN : hôtes dont le nom ne peut pas être résolu.

KNOWN : hôtes dont le nom peut être résolu.

PARANOID : hôtes dont le nom ne peut être résolu ou dont l’IP n’a pas de résolution inverse.

EXCEPT : permet d'exclure certains hôtes.

ALL EXCEPT poste10

/etc/hosts.allow est lu en premier, puis **/etc/hosts.deny**. La recherche s'arrête à la première correspondance trouvée. Une ligne dans hosts.allow autorise la connexion. Une ligne dans hosts.deny interdit la connexion. Si l'accès n'est pas explicitement refusé, la connexion est autorisée : la requête ne correspond à aucun critère.

Dans l'exemple suivant :

Seuls les membres du sous-réseau 192.168.1.0 ont le droit de se connecter au serveur ftp (interdit pour tous les autres).

Les hôtes poste1 et poste2 ont accès à telnet et portmap.

Les hôtes de baddomaine.org, sauf trusted, n'ont aucune connexion possible.

Le serveur pop/imap est interdit à tous ceux du réseau 192.168.0.0 sauf 192.168.1.5.

```
# /etc/hosts.allow

vsftpd:          192.168.1.

in.telnetd, portmap:    poste1, poste2



# /etc/hosts.deny

ALL : .baddomaine.org except trusted.baddomaine.org

vsftpd,in.telnetd,portmap : ALL

dovecot : 192.168.0. EXCEPT 192.168.1.5
```

4. Netfilter

a. Présentation

Netfilter est une architecture de filtrage des paquets pour les noyaux Linux 2.4 et 2.6. Le filtrage se fait au sein même du noyau au niveau des couches 2, 3 et 4 du modèle OSI, c'est-à-dire les liaisons données, réseau et transport. Il est par exemple capable d'agir à bas niveau sur les interfaces ethernet (2), sur la pile IP (3) et sur les protocoles de transport comme TCP (4). Le filtrage est **stateless** : comme Netfilter n'inspecte que les en-têtes des paquets, il est extrêmement rapide et n'entraîne pas de temps de latence.

L'inspection des contenus des paquets (protocoles applicatifs) peut se faire à l'aide d'extensions mais devrait cependant être réservée à des outils de l'espace utilisateur.

Autrement dit, Netfilter est un firewall agissant au niveau du protocole.

Le programme utilisateur permettant d’agir sur les règles de filtrage est **iptables**.

L’implémentation au niveau du noyau est réalisée par des modules.

b. Vie d’un paquet

Plutôt qu’un long discours, voici un schéma. Le paquet arrive par le haut et ressort par le bas. Entre les deux, il passe par différents états au niveau de netfilter.

Entrée du paquet

PREROUTING

Décision de routage

INPUT

Processus de routage

OUTPUT

routage

FORWARD

POSTROUTING

Sortie du paquet

Les étapes de la vie d'un paquet réseau avec netfilter.

Chaque état (rectangle) correspond à un point de filtrage possible par la commande **iptables**.

PREROUTING : traite les paquets à leur arrivée. Si un paquet est à destination du système local, il sera traité par un processus local (INPUT, OUTPUT). Sinon, et si le forwarding est activé, les règles FORWARD et POST_ROUTING seront appliquées.

FORWARD : les paquets ne font que traverser le système local. Traite les paquets routés à travers le système local.

INPUT : traite les paquets destinés au système local, en entrée (après le routage).

OUTPUT : traite les paquets quittant le système local, avant POSTROUTING.

POSTROUTING : traite les paquets juste avant leur sortie du système.

c. Principe des règles

Lorsqu'un paquet est traité par netfilter, il l'est par rapport à un certain nombre de règles qui déterminent ce qu'il faut en faire.

Les règles sont ordonnées : la position d'une règle dans une liste indique quand et si la règle sera utilisée.

Les paquets sont testés avec chacune des règles, l'une après l'autre.

Netfilter fonctionne selon le mécanisme de la première correspondance. Si une règle correspond, les autres règles sont négligées et la vérification s'arrête.

Une règle peut spécifier plusieurs critères.

Pour qu'une règle corresponde à un paquet, tous les critères doivent correspondre.

Si malgré toutes les règles, le paquet passe, une règle par défaut peut être appliquée.

d. Cibles de règles

Une cible de règle détermine quelle est l'action à entreprendre lorsqu'un paquet correspond aux critères d'une règle. On utilise l'option **-j** de **iptables** pour spécifier la cible.

Les deux cibles de base sont DROP et ACCEPT. Des extensions de netfilter ajoutent d'autres cibles comme LOG ou REJECT.

DROP : le paquet est rejeté. Aucune notification n'est envoyée à la source.

REJECT : le paquet est rejeté, retournant une erreur à la source.

ACCEPT : le paquet est accepté.

LOG : une information est envoyée à syslog pour les traces.

Vous pouvez créer des règles sans cible. Dans ce cas, la règle incrémentera un compteur de paquets et un compteur d'octets associés à la règle afin d'établir une collecte de statistiques.

e. Premier exemple

Voici une règle simple qui interdit tous les paquets en provenance de 192.168.1.11.

iptables -A INPUT -s 192.168.1.11 -j DROP

-A : point de filtrage (INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING) qu'on appelle aussi **chaîne**.

-s : source, peut être une adresse IP, un nom d'hôte, un réseau, etc.

-j : jump, cible de la règle (ACCEPT, DROP, REJECT...)

Résultat : vous interdisez l'entrée des paquets dont la source est 192.168.1.11.

f. Opérations de base

Les règles sont numérotées à partir de 1.

Ajoutez une règle avec **-A**.

iptables -A INPUT -s 192.168.1.2 -j DROP

Insérez une règle avec **-I** à la position souhaitée.

iptables -I OUTPOUT -d 192.168.1.25 -j DROP 3 # inserer à la

3eme position

Supprimez une règle avec **-D**.

iptables -D INPUT 1 # supprime la règle 1

Listez les règles avec **-L**.

iptables -L OUTPUT

iptables -L

Chain INPUT (policy ACCEPT)

target   prot opt source   destination

Chain FORWARD (policy ACCEPT)

target   prot opt source   destination

Chain OUTPUT (policy ACCEPT)

target   prot opt source   destination

Utilisez **-F** (flush) pour supprimer l'ensemble des règles.

iptables -F

Utilisez **-P** (policy) pour modifier les règles par défaut d'une chaîne.

iptables -P INPUT DROP

iptables -L INPUT

Chain INPUT (policy DROP)

target   prot opt source   destination

g. Critères de correspondance

Général

Les critères de correspondance déterminent la validité d'une règle. Tous les critères doivent être vérifiés pour qu'un paquet soit bloqué. Les critères de base sont :

-i : interface entrante (filtrage couche 2) ;

-o : interface sortante (filtrage couche 2) ;

-p : protocole de couche 4. Pour les noms, voir le fichier **/etc/protocols** ;

-s : adresse IP de la source (ou réseau) ;

-d : adresse IP de destination (ou réseau).

Interdire les entrées par eth0.

iptables -A INPUT -i eth0 -j DROP

Interdire le forward entre eth1 et eth2.

iptables -A FORWARD -i eth1 -o eth0 -j DROP

Interdire le protocole ICMP en entrée (le ping !).

iptables -A input -p icmp -j DROP

TCP, UDP et ICMP

Suivant le protocole (couche 4) certaines options sont possibles. C'est le cas de tcp, udp ou icmp (notamment utilisé par ping). Le filtrage au niveau des protocoles est généralement effectué par des extensions à netfilter.

-p : protocole (tcp, udp, icmp, etc.)

--sport : port source

--dport : port destination

Si vous souhaitez par exemple interdire les connexion entrantes à destination du port 80 (serveur httpd) procédez ainsi :

iptables -A INPUT -p tcp -dport 80 -j DROP

Arguments des critères

Pour les adresses, vous pouvez spécifier :

un hôte par son nom ou son adresse IP ;

un réseau par son nom ou son masque (192.168.1.0/24, 192.168.1.0/255.255.255.0).

Pour les ports :

un numéro ;

un nom (voir /etc/services) ;

une gamme de ports : **123:1024**.

Dans tous les cas, la négation avec ! (point d'exclamation) est possible.

Pour interdire en entrée toutes les connexions sauf celles de 10.0.0.1 :

iptables -A INPUT -s ! 10.0.0.1 -j DROP

h. Sauver ses réglages

Les règles définies avec iptables sont chargées en mémoire et transmises dynamiquement au noyau. En redémarrant la machine, elles sont perdues. **Red Hat** permet de sauver l'ensemble des règles de manière à les rendre persistantes.

service iptables save

Les règles sont sauvées dans le fichier **/etc/sysconfig/iptables**.

Pour les autres distributions, consultez la documentation. Dans certains cas, il peut être utile de créer son propre script.

Les règles iptables devraient être chargées **AVANT** l'activation du réseau. Ainsi il n'y a aucun risque au niveau de la sécurité car les règles seront directement valides à l'activation du réseau.

TP : La sécurité : Sécurité générale du système & Sécurité réseau

114.3 Sécurité de l'utilisateur (1)

Voire 114.1 Tâches d'administration de sécurité (4)

Nom du document : cours_lpi102_version_eleves.doc
Répertoire : D:\Farid\Partner_formation\Preparation_Cours\Linux
Modèle : C:\Documents and Settings\Fulgore\Application Data\Microsoft\Modèles\Normal.dot
Titre : PREPARATION
Sujet :
Auteur :
Mots clés :
Commentaires :
Date de création : 26/03/2009 22:48:00
N° de révision : 51
Dernier enregistr. le : 28/03/2009 19:11:00
Dernier enregistrement par : LSD
Temps total d'édition :229 Minutes
Dernière impression sur : 27/07/2009 09:47:00
Tel qu'à la dernière impression
Nombre de pages : 276
Nombre de mots : 56 230 (approx.)
Nombre de caractères : 309 266 (approx.)